Department of Mechanical Engineering

FACULTY OF ENGINEERING AND DESIGN

FINAL YEAR BEng PROJECT REPORT (ME30227)

Development of an Electric Motor Test Rig (dynamometer) for TBRe (ID 342)

Oliver Allin

07/05/2024

Word count: 6798

"I certify that I have read and understood the entry in the Student Handbook for the Department of Mechanical Engineering on Cheating and Plagiarism and that all material in this assignment is my own work, except where I have indicated with appropriate references."

Author's signature: ……..

Supervisor: *Christopher Vagg*

Assessor: *Roger Ngwompo*

Summary

This investigation explores the design and development of a dynamometer for the Team Bath Racing electric team at the University of bath. This report focuses on the data logging aspects of the test rig. Designing and programming an Arduino to record and display data relevant to the characteristics of the motor under test. The main characteristics being torque (Nm) and motor speed (RPM). The objectives of this investigation are as follows: To produce a data logger that is accurate in reading newton loads measured by load cells on the dyno, The data logger to accurately measure speed utilising a hall effect sensor, Data to be recorded so that it is easy for post-test manipulation and analysis, and the data logger to be highly accessible and easy to use by all individuals. In order to achieve these objectives, a series of stage gates were created outlining major developments in both the data logger's hardware and software. The stage gates being a basic test, a display test, a double load cell test, and a final test. The final test produced the results needed to accurately compare the torque and motor speed of the UUT to the requested values of RPM by the controlling VESC software and expected torque values assuming 100% motor efficiency. Comparing these values allowed the accuracy of the data logger to be calculated. The results showed that at a motor speed of over 500 RPM, a 0.05% error in the RPM reading and an 8.71% error in the torque reading obtained by the data logger was found. This investigation concludes that the data logger is successful in accurately outputting and storing the relevant data required from the dynamometer. The design of the software also allows for further future improvements and refinements.

Acknowledgments

I would like to acknowledge the following people, whose contributions and support have been vital to the completion of this project:

Dr Chris Vagg

Ryan Hughes

Alexandre About

Ibraheem Rodrigues

Alex Cucchiara

Eugene Levinson

And the members of the Team Bath Racing electric team

Table of Contents

Table of Figures

Nomenclature

UUT	Unit Under Test
TBRe	Team Bath Racing Electric
Dyno	Dynamometer
RPM	Rotations Per Minute
TFT	Thin Film Transistor
USB	Universal Serial Bus
ESB	Emergency Stop Button
MDF	Medium Density Fibreboard
PCB	Printed Circuit Board
ERPM	Electrical Rotations Per Minute
DC	Direct Current
UoB	University of Bath

1 Introduction

This project consists of the design and development of a data logging system to accurately measure and output data for the Team Bath Racing electric team (Team Bath Racing electric, n.d.). to allow the team to investigate and develop their motor selection for their competition car.

1.1 Project Background

1.1.1 Formula Student

TBRe are a formula student team (Institution of Mechanical Engineers, 2024) competing from bath university. Formula student is a Europe wide engineering competition that encourages young engineers to innovate and develop enterprise. The University of bath being one of the foremost Electric Formula Student teams in the UK, they need to be selecting the most powerful, cost effective and light-weight motor as possible.

Figure 1 - TBRe 2023 Car (Team Bath Racing electric, n.d.)

This year the team identified a new motor that could be used in future cars which becomes our Unit under test for the dyno. This being a 25KW brushless Motor (FREERCHOBBY, n.d.) for $370.00 USD seen in figure 2. With similar alternative motors such as the 20-55kW motor by Electric Motorsport costing $1,170.00 USD, this would be a massive reduction in cost. However, with the UUT claiming to have a power output of 25KW producing 35Nm worth of torque, the team needs to validate these claims before they are able to commit to purchasing them for the competition car. Currently the university has a few dynamometers available for use by faculty, researchers, and students upon request. These dynamometers have a much higher specification and are used for testing larger motors and longer drive cycles. Due to this they require a university technician to be present when the test rig is operating. In addition to this, other research that is conducted at the university may take priority over TBRe leaving them unable to test their motors to the full extent they require. Therefore, the team requires a dyno in which they have sole use over to allow them to test new and pre-existing motors to their full extent without the need for a university technician or prescheduled appointments.

Figure 2 - 25KW Brushless Motor from FREERCHOBBY (FREERCHOBBY, n.d.)

Figure 3 - 20 kW - 55 kW Brushless Motor Competitor (Electric Motorsport, 2016)

1.1.2 Project Layout

This project is a joint project with a University of Bath Masters student, Alexandre About, who is in-charge of the mechanical and high voltage aspects of the dyno. This report investigates the design and development of the data logging system. We collaborated alongside each other to test the new motor and develop the dynamometer so that it is easily used. Alexandre was also in-charge of selecting the load cells and hall effect sensor due to timing conflicts between the BEng and MEng modules.

1.1.3 Pre-existing Dynamometer

Upon the introduction of the project in November 2023, the skeleton of a previous dyno project was handed over to us which can be seen in Figure 4. Consisting of very basic framework and no accurate and effective way to measure and display key dynamometer values such as torque.

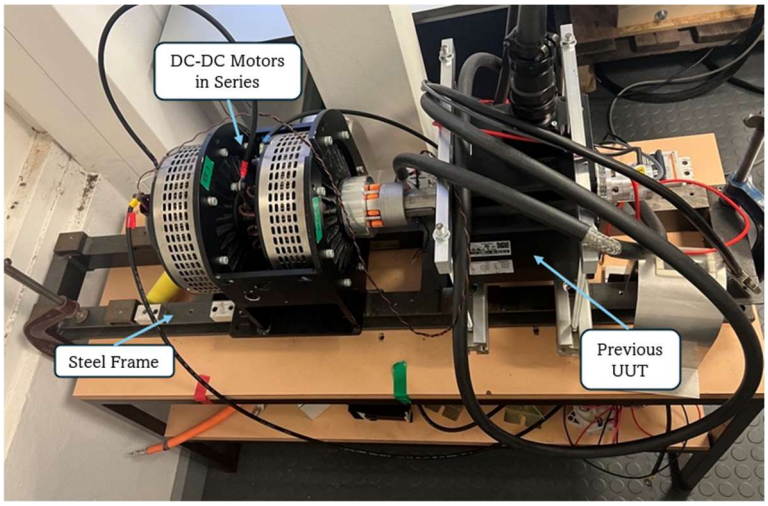

Figure 4 - Pre-existing Dyno handed over 8/11/23

1.2 Aims and Objectives

1.2.1 Overarching Aim

"Upon completion of the project, TBRe must be equipped with a functional and easily accessible 'dynamometer' which will measure, and output data related to hub-motor characteristics, with scope to be used for the validation of motor control algorithms. "

This report focuses on the investigation into the measuring and outputting of data, ensuring it is easily accessible and accurate. Due to the time limitation of the investigation, the software and electrical aspects of the dyno have been designed in a way that allows for easy future development.

1.2.2 Key Objectives

To achieve the aim of the project, software and electrical specific objectives were created.

- The Arduino will be capable of interfacing with the load cells, hall effect sensor and touch screen to record and display data at a high frequency and accuracy.
- The software will be able to store data from the sensors on an SD card. The data will be correctly formatted and stored for easy evaluation post-test.
- The system will be able to accurately produce live readings of Torque and RPM. Recording up to 7000 RPM.
- All electrical systems will be safely and efficiently stored in a casing which is durable and aesthetically pleasing, with tidy and secure electrical connections inside.

1.2.3 Future Objectives

A few objectives that will be explored and developed following on from this report:

- To control the UUT via the display or a preloaded drive cycle.
- The data logger will also be able to vary the resistive torque on demand.

2 Literature review

2.1 Dynamometers

2.1.1 History

Dynamometers have been used as a means to calculate force dating back as early as 1798 in Paris by Edme Régnier (Horne & Talbot, n.d.). The Régnier device was used to measure the rolling resistance produced by horse driven wagons. Dynamometers then split into two different fields of investigation: The first being Anthropology and the second being Mechanical Work measurement (Loude, 2014).

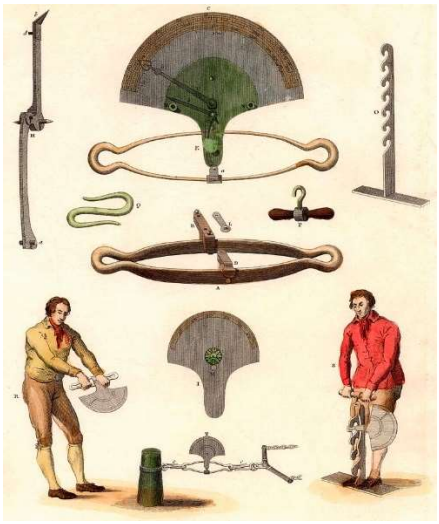

Figure 5 - Regnier Dynamometer (Archive/uig, 2016)

Nowadays, there are 5 main types of dynamometers that are widely used: Eddy-current Dynamometers, Electric Dynamometers, Water-brake Dynamometers, Prony Brake and Fan Brakes. Each dynamometer has it benefits and drawbacks; Eddy-current and water break dynos being relatively affordable as opposed to an electric dynamometer which is the most applicable but with the highest cost to manufacture (Romandoni, 2021).

2.1.2 Electric Dynamometers

An Electric dynamometer is a device used to measure the torque, force, speed, and power required to drive a motor. To measure these values, the torque and rotational speed (Industrial Quick Search, n.d.) (RPM) of the test rig are taken. An Electric dynamometer is a form of absorption dyno, it creates a resistive torque on the shaft via an additional motor. Electric dynos can test various forms of drive trains, from combustion engines to electric motors. They are an accurate way to provide torque and power readings of a motor before it is selected for use.

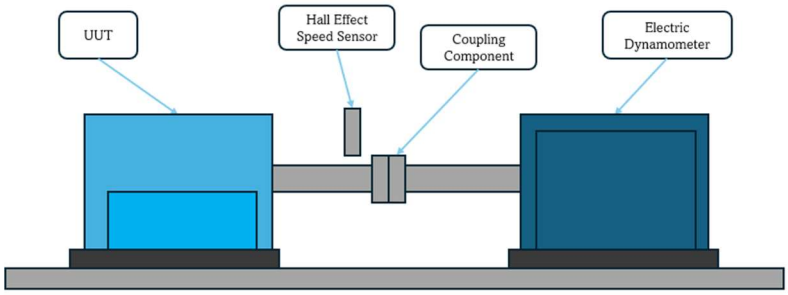

Figure 6 - Example Diagram of an Electric Dynamometer

2.2 Sensors

2.2.1 Load Cells

In order to calculate the torque of the UUT, the force moment being applied by the motor needs to be measures. To do this, 2 load cells are used either side of the UUT. A load cell is a device that measures a directional force, this could be compression or extension. (RS, n.d.)The load cells used in this investigation is a strain gauge load cell. It is composed of a material of high strength with slight elastic properties allowing some deformation under pressure. The strain gauge within the load cell can measure this deformation producing a value which can be converted into a weight value or Newton force.

Figure 7 - Load Cell used on the Dyno (Pushton Electronic Equipment, n.d.)

The load cell selected for the TBRe Dyno was an Alloy Steel load cell from Pushton Electronic that can record a range up to 300kg. This was selected as it correlated to the maximum torque TBRe expect the motor to perform at whilst still allowing for accurate measurements in the lower regions of force.

2.2.2 HX711

To process the data off the load cells, HX711 amplifiers are required. A HX711 amplifier is a 24-bit Analog-to-digital converter. It is used to amplify the signal from the load cell and reporting them to a microcontroller, in this case that is the Arduino. (Yida, n.d.)

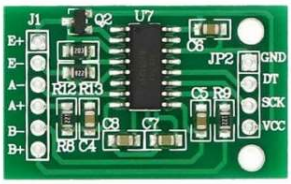

Figure 8 - HX711 Amplifier (Quartz Components, n.d.)

2.2.3 Speed Sensor

To measure the rotational speed of the UUT, a speed sensor is required. This investigation utilises a hall effect sensor which is made up of a semiconductor material, which allows a voltage change when the material is placed in a magnetic field (RS, n.d.). The Hall effect speed sensor works by outputting a digital waveform, pulsing to around 5 volts when a tooth of the wheel passes by the hall effect sensor. The amplitude of the signal always remains constant, but the frequency of pulses increases as RPM increases. The time difference between the pulses is what is required to calculate the RPM of the wheel.

Figure 9 - Speed Sensor Hall-Effect Mini-HA-P (BOSCH, n.d.)

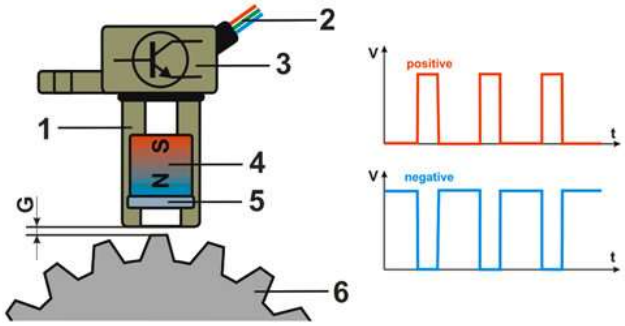

Figure 10 - Hall Effect Speed Sensor (Mucevski, 2015)

1. Sensor housing, 2. Output wires (+Vcc, −Vcc and signal), 3. Integrated electronics, 4. Permanent magnet, 5. Hall Effect device, 6. Trigger wheel, G. Air gap

3 Methodology and Experimentational Method

3.1 Dynamometer Set Up

The Dynamometer was mechanically built prior to this investigation. The layout of the dyno can be seen in figure 13. The Dyno is split into 3 separate circuits: The UUT, The resistive torque and the data logging. The UUT layout is comprised of the motor being tested mounted on a back plate which sits on two load cells. Figure 12 demonstrates the layout in which the load cells are placed. The UUT is powered by a 24V Lithium-Ion battery. Its power is controlled by a TRAMPA VESC 100/250 controller (TRAMPA BOARDS, n.d.) communicating to an external laptop. The layout of this circuit is represented in figure 14.

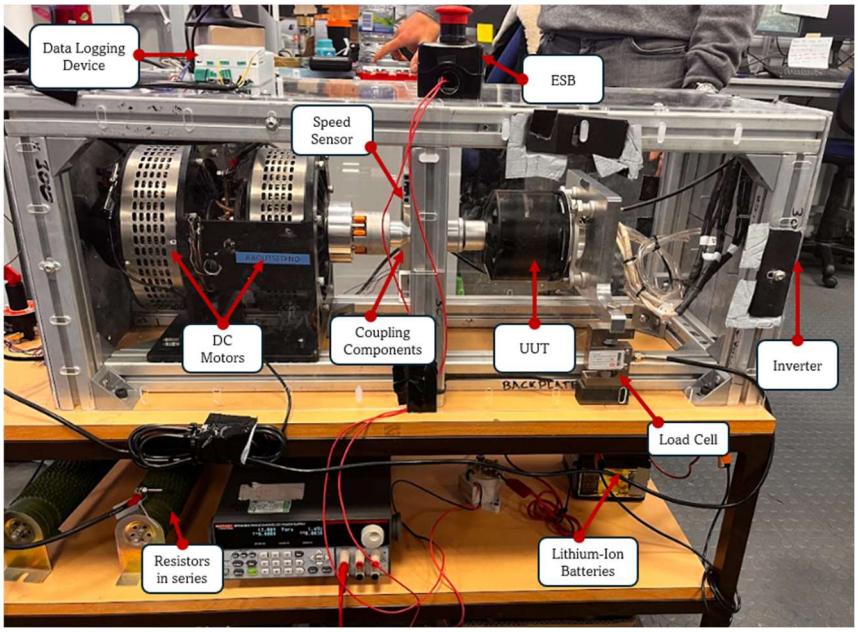

Figure 11 - Labelled Image of the Dyno set up.

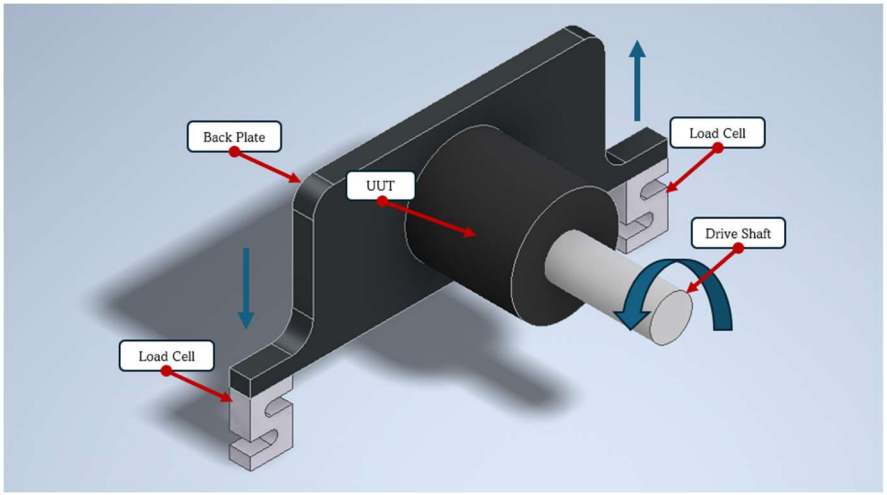

Figure 12 - Load Cell Placement to Measure Torque of the UUT

The resistive torque circuit is made up of 2 DC motors connected in series with two resistors of 2.2Ω and 2.6Ω. There is a switch in place to vary the load whilst running cycles. Figure 13 demonstrates how the resistive torque circuit was set up for this investigation.

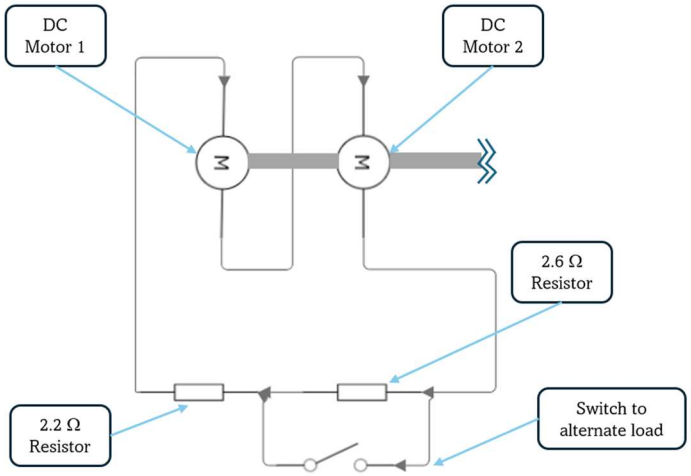

Figure 13 - Electrical Layout of Resistive Torque Circuit.

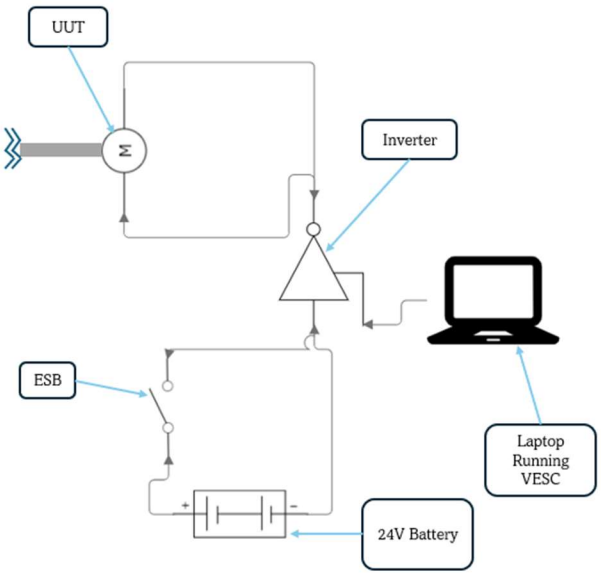

Figure 14 - Electrical Layout of UUT Circuit.

The final circuit being the one that this report invloves is the data logging circuit. Figure 17 shows the electrical layout of how the load cells are connected through the HX711 Amplifiers into the arduino, along with the hall effect sensor and the TFT LCD Display. Each connection being soldered or connected via male/female pins in order to maintain continuity and avoid any electrical noise in the system. The arduino can be powered by either a usb port, a wall socket adapter or a powerbank. The programming to display and record data of the dyno is preprogrammed onto the arduino.

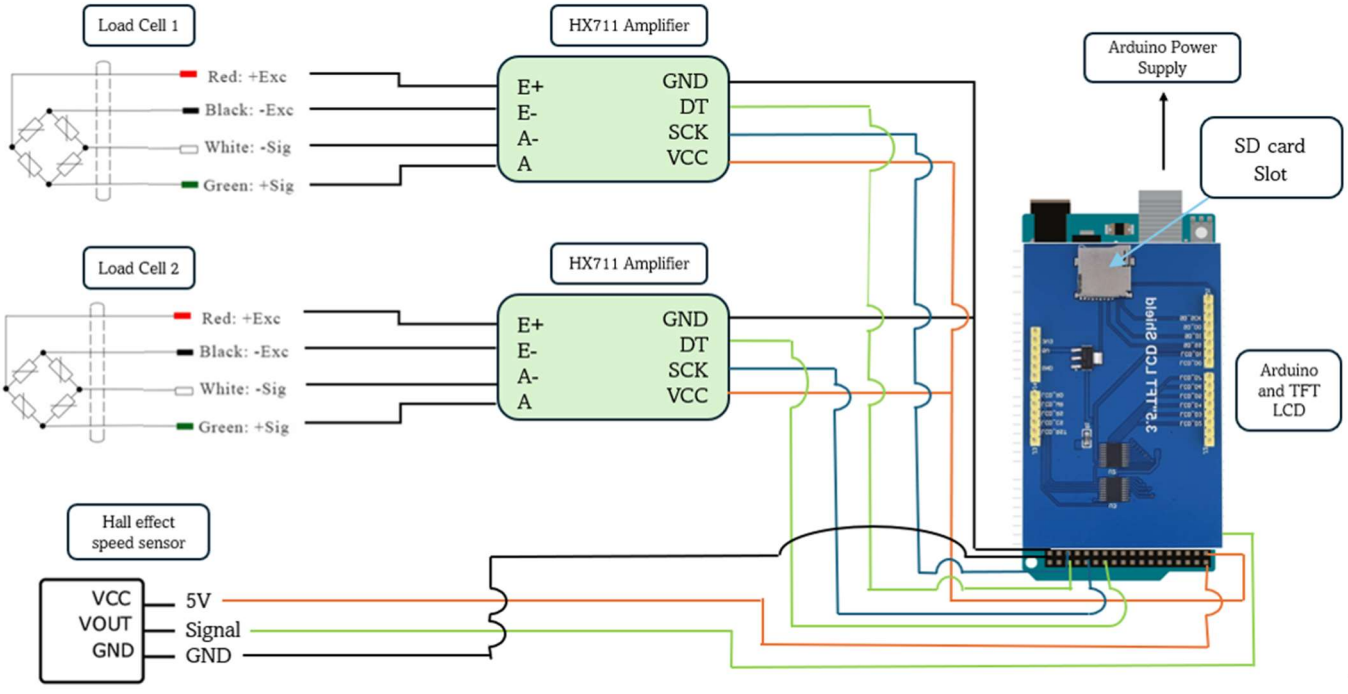

Figure 15 - Electrical Layout of Data Logging System.

3.1.1 Arduino Selection

This investigation utilises the abilities of an Arduino Mega2560 board. It is a microcontroller with 54 digital pins, 16 analog inputs and a USB connection port (Arduino, n.d.). The Arduino allows the signals produced by the hall effect and load cells to be processed and outputted to the user. The Mega2560 was selected for the dyno as it has sufficient pins to mount the touch screen directly onto the board, Reducing the total space it takes up.

Figure 16 - Arduino Mega2560 (Arduino, n.d.)

3.1.2 Touch Screen

In order to display the values in real time without an additional computer, a TFT LCD screen was selected. A TFT display was chosen due to its low power consumption, fast and accurate response time and it being a space efficient design. (Ryan, 2021)

Figure 17 - TFT 3.5 inch LCD screen (Amazon, n.d.)

3.2 Development Stage Gates

Due to this investigation being partnered with another student, A Gantt chart was developed. Intertwining the universities appointed deadlines to allow time for report writing and key stage gates created to progress the aims and objectives stated previously.

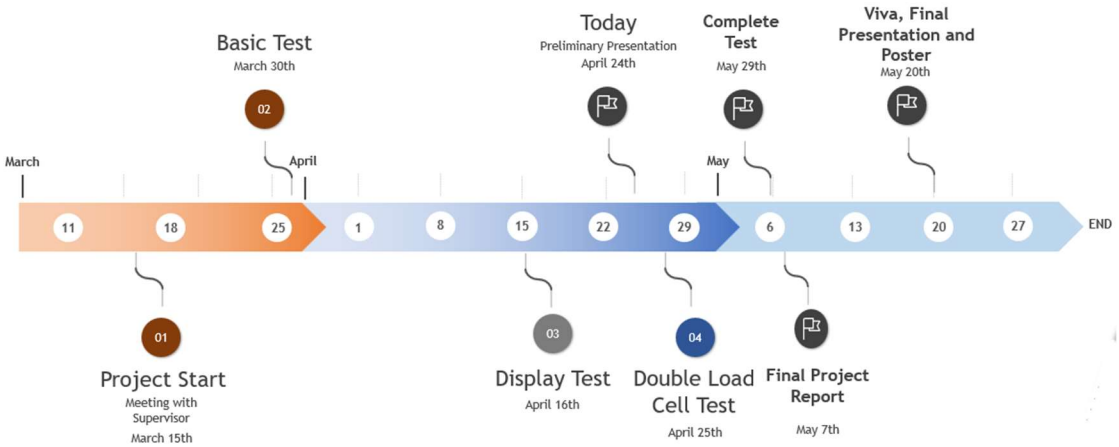

Figure 18 - Gantt Chart of Project Timeline.

The project was split up into 4 main stage gates: A basic test, A display test, A double load cell test and a complete test. These gates were decided due to them each being a step up in both the hardware and software elements of the data logger. These stage gates also allowed the other student to test their aspects of the dyno before their deadlines.

3.2.1 Basic Test

The first stage gate was a basic test which comprised of a single load cell and hall effect speed sensor sending data through the Arduino to be read by MATLAB. This test allowed for a proof of concept that the load cells were successfully outputting data that could be read and displayed. Figures 19 and 20 are examples of the data that was recorded for the test. The program created to display this data can be found in the appendix (Appendix 1).

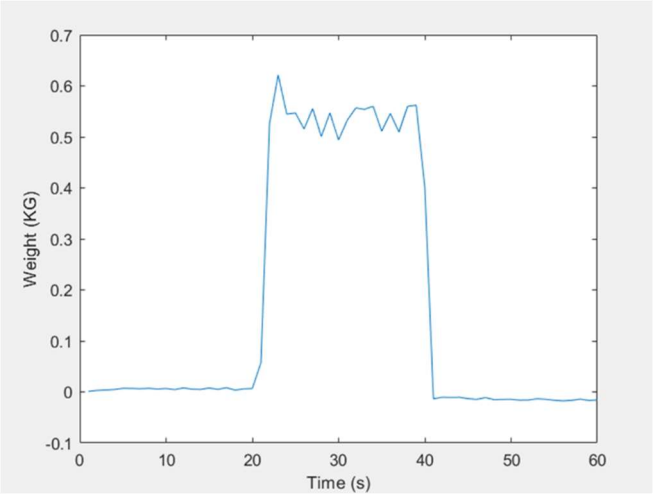

Figure 19 - Basic Test Results Single Loading.

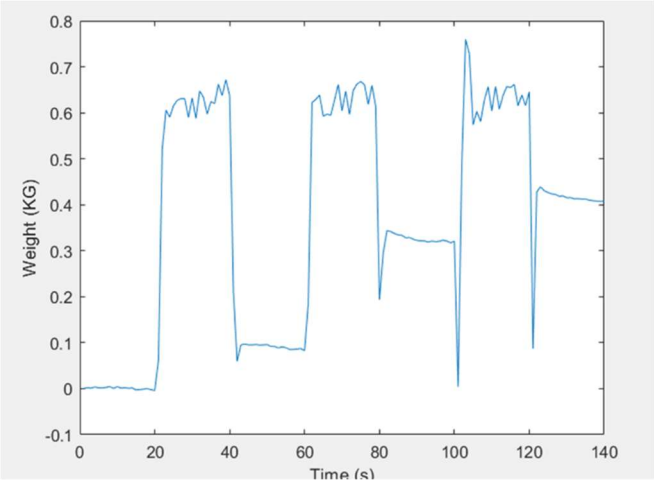

Figure 20 - Basic Test Results Loading and Unloading.

Upon completion of the basic test, it was established that MATLAB would not be able to process signals fast enough to accurately read rpm. This led to the learning and investigation into programming through Arduino IDE. Allowing for quicker data processing and eliminating the means for the additional laptop.

3.2.2 Display Test

The next stage gate was to begin displaying live data on the TFT display. Through a verification and validation method of testing, running the dyno with the data logger plugged in and comparing the values being displayed to what is being requested by the VESC tool controlling the UUT. At this stage design of the housing had begun and a basic case was made to reduce any cable noise and tidy the data logging system. At this stage it was also decided that the addition of another load cell was required. This was to account for reaction forces experienced within the dyno and to reduce any inaccuracies.

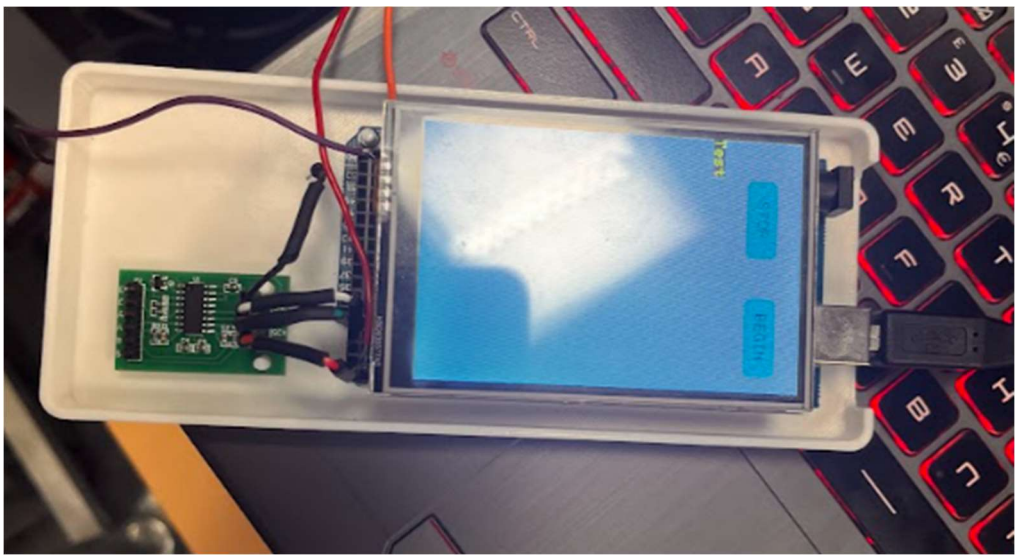

Figure 21 - Display Test Displaying Buttons and Test Data.

3.2.3 Double Load Cell Test

Once the second load cell arrived, it was implemented into the software and the logger began displaying live readings for RPM and loads. Now the data needed to be calibrated to ensure that the values that they were displaying were accurate, meeting the objectives and aim for this development project. For the RPM it was again a case of comparing the displayed value to the requested RPM.

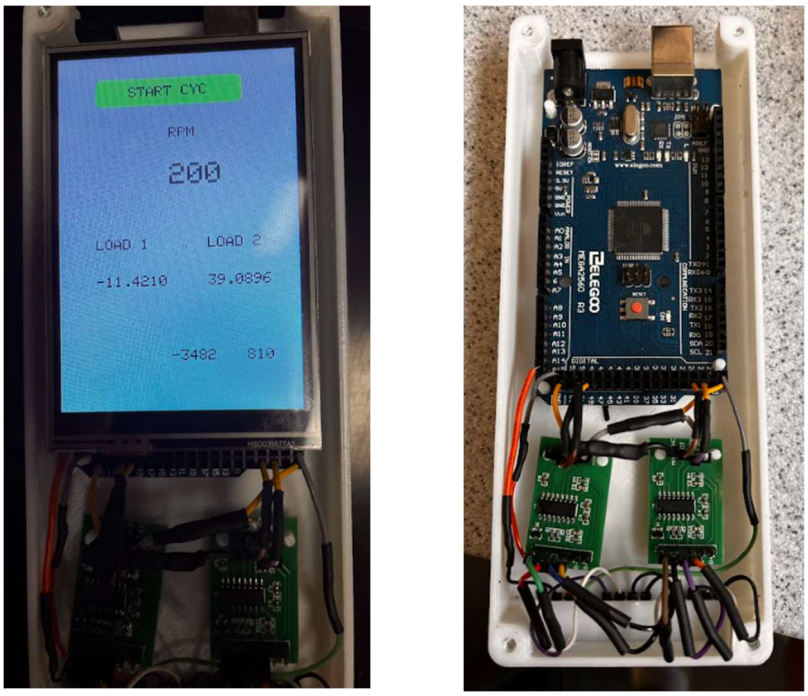

Figure 22 - Double Load Cell Test Layout

The load cells required additional calibration due to the amplifiers. To get an accurate value for force acting on the load cell, a scale factor had to be established. This was done by taking a tare weight of the load cell (T), subtracting this from the load cell reading with a known mass on the load cell (P) and dividing it by the value of the known mass (P_t). For this investigation the load cells needed to read in newtons. To do this, Equation 1 was adjusted by the known mass multiplied by gravity (9.81).

$$f_{scale} = \frac{P-T}{P_t} \qquad [1]$$

This was repeated for each of the load cells. The formula to then display the newton loads on the display was calculated using equation 2. With the tare value (T) being taken at the start of every cycle, and the scale factors (f_{scale}) being a set value for each sensor.

$$Load = \frac{P-T}{f_{scale}} \qquad [2]$$

Once the load cells were calibrated, they were introduced into the dyno for further calibration. A loading arm was manufactured from MDF, as seen in figure 23, to be attached to the rear of the UUT plate. Doing this allowed a set mass to be applied to arm and hence a known load to be experienced by the load cells. This set up is shown in figure 24, with a known mass attached to the arm it was a matter of hand calculating the torque being applied {See Alexandre's Report for Further details} and comparing this value to the loads that the data logging was reading.

Figure 23 - MDF manufactured for calibration.

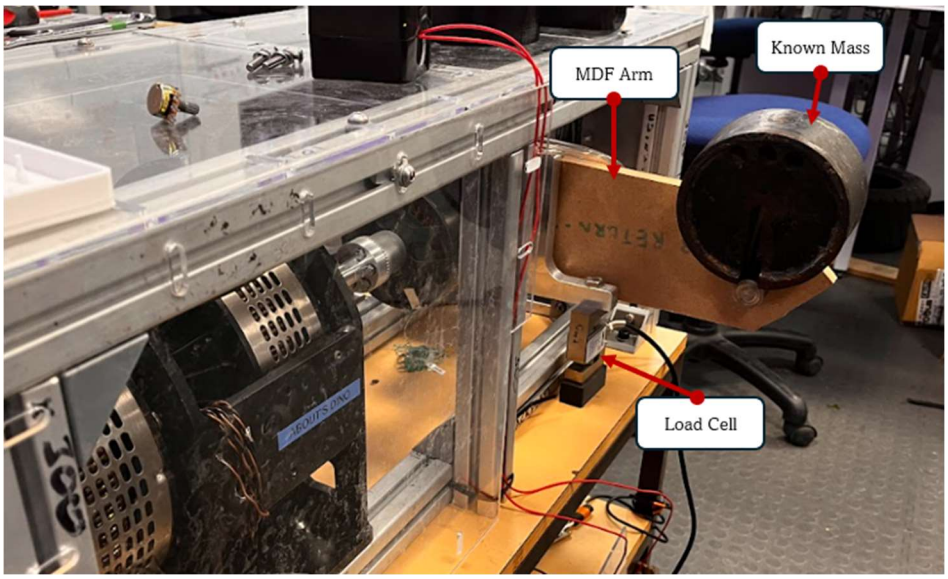

Figure 24 - Dyno set up for load cell calibration.

3.2.4 Complete Test

The final test was to run a drive cycle with varying RPM and measuring the torque experienced by the UUT. Recording the RPM and loads at 0.5 second intervals, a minute long drive cycle was started. The Data logger was powered via a USB port connected to a laptop, and it was displaying the live readings whilst also recording them on the laptop for further data processing post-test.

Figure 25 - Data logger set up for final test.

2 different tests were conducted, A load on test and a no-load test. Both tests consisted of increasing the RPM from 0 to 571 RPM in 4 equal increments (10 seconds for each RPM) then returning to 0 at the end of the test. These tests were repeated twice to record any trends and avoid anomalous results in the data logging.

At this stage the data was only recorded onto the laptop in a temporary manor as an SD card had not been found to record the data onto. The software included the means to record the data on the SD card at this time, it was just unavailable.

3.3 Arduino Housing

3.3.1 Basic Design

When the project began, it became apparent that a housing would need to be design for the Arduino and other relevant electrics for the data logger. It can be seen in Figure 26 that the Arduino was lose and free which caused inaccuracies in the results due to noise experienced in the cables and connections.

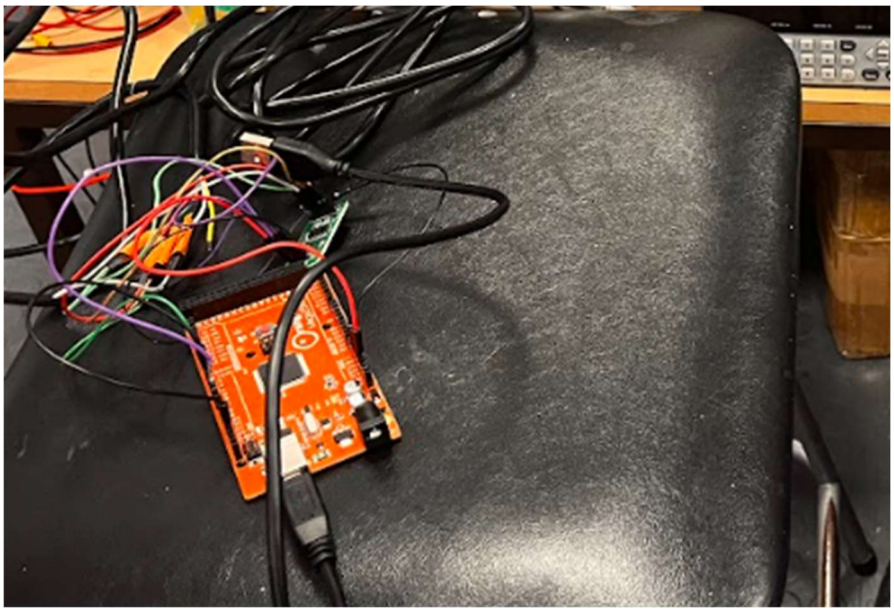

Figure 26 - Basic Test Arduino Layout

A basic casing for the Arduino was designed to hold it and the amplifiers in place while tests are conducted. It was designed without a cover initially for easy access to the components and wiring. An access port was included to ensure the Arduino can still be powered. The base consists of 4 aligning columns for the Arduino and 2 for the HX711 amplifier, along with a pillow block to support the amplifier.

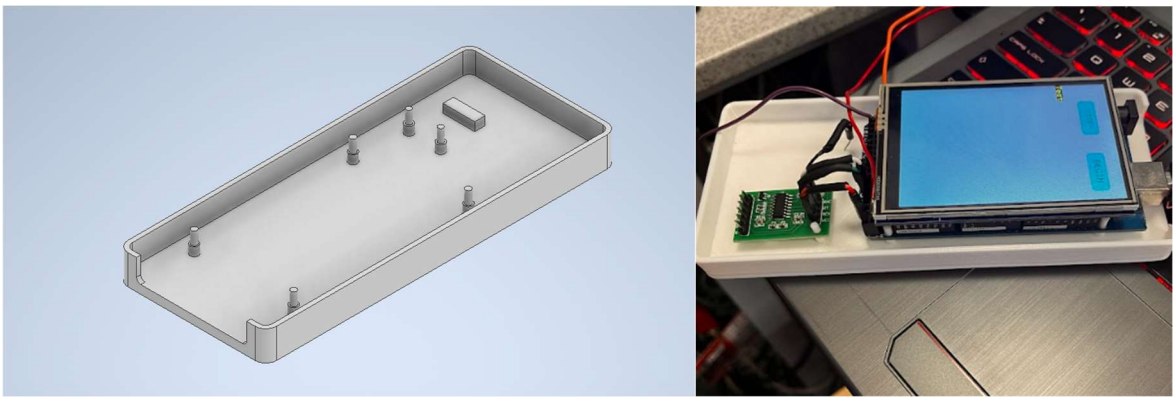

Figure 27 - Arduino housing with single amplifier.

3.3.2 Double Load Cell

The next design iteration came when the new load cell was added. This meant extending the length of the case and creating another mount for the second HX711 Amplifier. At this stage the wires connecting the Arduino to the amplifiers were cut and soldered to bespoke lengths to ensure a compact design.

Figure 28 - Double Load Cell Casing

In addition to the new amplifier mount, captive nuts were added in each corner. This meant that the base wouldn't require a reprint when the cover was designed, and the final design can be secure.

3.3.3 Casing Cover

The final addition to the case design was the case cover. This was designed to ensure that only the touch screen was accessible without needing to open the case, ventilation was added to the sides of the cover to ensure the hardware wouldn't overheat on long durations tests.

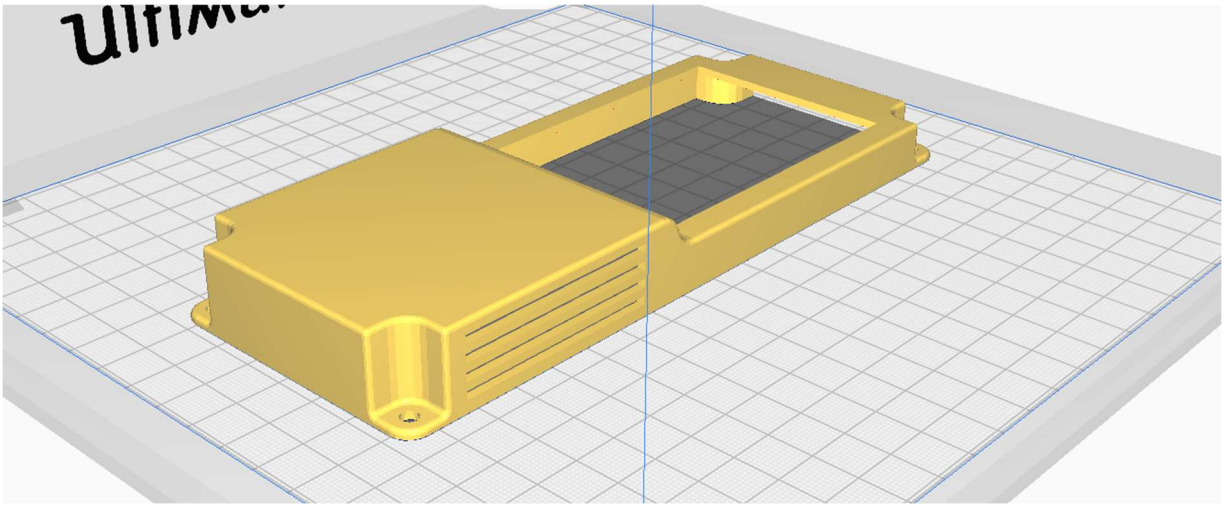

Figure 29 - 3D CAD model of the case cover.

4 Results and analysis

4.1 Final Case Design

The final design of the case was 3D printed in 2 parts: a base and a cover. The base and cover are connected using 4 M3 bolts and captive nuts in the base ensuring a secure fit. The Arduino ports are accessible from the top of the device and the female input ports for the load cells and speed sensor are available at the bottom.

Figure 30 - Final Case Design

4.1.1 Design for Accessibility

The case has been designed so that the only set-up required by the individual using it, is to plug the Arduino in and plug the relevant sensors into the reader. Basic instructions of which coloured wire is assigned to which port is labelled on the outside above the relevant ports.

Figure 31 - Sensor Port on Data Logger.

Another design feature that ensures the user has limited access to the electronics is the cover. Only giving access to the touch screen, it prevents the user from removing or damaging the internal components.

4.1.2 Electrical Safety

3D printing was selected as the production method for the case due to its ability to create complicated structures in addition to its thermal properties. Depending on the duration of the drive cycle the dyno is running, the Arduino and electrical components tend to increase in temperature. The 3D printing material has a high heat resistance, ensuring that the user will not encounter any discomfort when using the device or the device will not damage itself during test, affecting the reliability of the device.

Figure 32 - Final Casing Design (Top view & Side View Respectively).

In addition to the heat resistant material, the case also has built in ventilation on the sides of the walls. Allowing air to flow in through the top of the case and out the side, dispersing any heat generated.

Another electrical safety feature of the case is the management of cables within. Every connection within the data logger was soldered, had its continuity checked and then added heat-shrink in order to ensure there was no shorting or means for unreliable results.

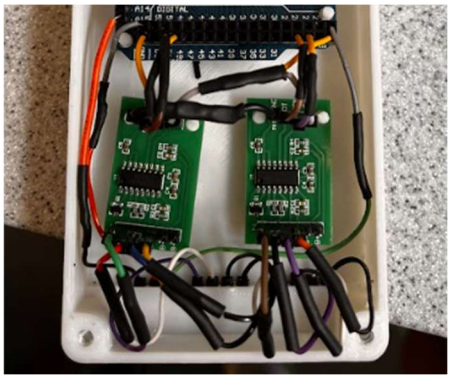

Figure 33 - Interior Cable Management.

4.1.3 Future Proofing

As this project was time limited, the design of the case was to ensure that it could be accessible for troubleshooting and future improvements. The use of M3 bolts allows an individual to remove the cover to inspect the electronics and make any required adjustments. Whilst this design is effective in achieving the objectives set out for this project, it falls short as it currently only has the space and available ports to read the 2 load cells and the speed sensor. For further improvement to the case, Additional open ports would be added to the base to allow space for more sensors such as a thermal sensor to measure the temperature of the UUT and control connections to directly control the UUT. In addition to this, I would mount the HX711 amplifiers onto a PCB board to reduce the height of the logger.

4.2 Software

The software for the data logger was developed in Arduino IDE. It can be split into 4 aspects: The Set Up/Libraries, The Live Reading/Display, Load, RPM and Torque readings, and the Data logging. The entirety of the code can be found in the appendix (Appendix C). To achieve the objective and aim of this project, the conde needed to be concise and efficient.

4.2.1 Set Up & Libraries

In order to set up and program the Arduino a number of libraries had to be installed. Libraries increase the functionality of the Arduino, allowing it to interact with additional hardware such as the load cells and the touch screen display. The additional libraries that were required to program the data logger are as follows:

Adafruit (Arduino, n.d.) – This library is used for the graphics on the display.

MCUFRIEND (Arduino, n.d.) – Additional graphics library working with Adafruit on the TFT Display.

TouchScreen (Arduino, n.d.) – The library required to interact with the touchscreen display.

HX711 (Arduino, n.d.) – The library needed to allow the arduino to interact with the amplifiers.

SD (Arduino, n.d.) – The library needed to enable reading and writing on the SD card.

These and additional preinstalled libraries are called at the start of the program as shown below:

```
#include <Adafruit_GFX.h>
#include <MCUFRIEND_kbv.h>
MCUFRIEND_kbv tft;
#include <TouchScreen.h>
#include <Adafruit_I2CDevice.h>
#include <HX711.h>
#include <SPI.h>
#include <SD.h>
```

Figure 34 - Library Installation Code

4.2.2 Live Readings & Display

To prevent unnecessary data being record when the Arduino is on, a “Begin” button was added. When this is pressed the screen will refresh and begin displaying live data at a refresh rate of 0.5 seconds. The program is currently set to run for 1 minute and will tare the load cells at the being of the cycle. To ensure accurate reading the dyno must be stationary when the test begins. During the test there are 5 pieces of information that are displayed live: the RPM, The force load (In Newtons) experienced by each load cell, the time since the beginning of the test (In Seconds) and the calculated torque.

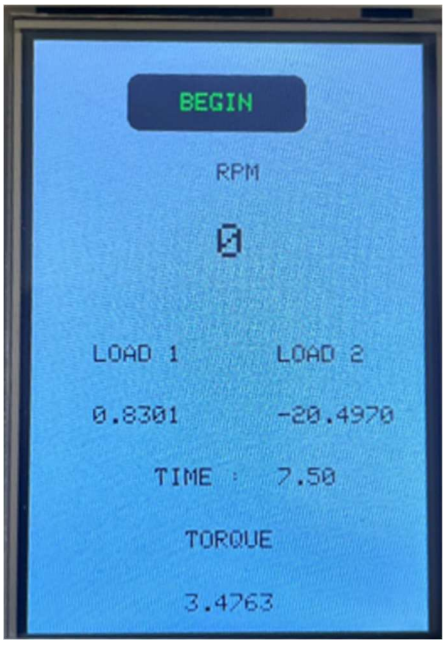

Figure 35 - Data Logger Display Layout.

The live data is displayed in order of importance to the drive cycle being run in this investigation. The UUT is requested a set RPM at different time intervals, hence the need to read a live RPM to ensure the requested RPM is equal. Following on from this the two loads are displayed. These are included to ensure that both load cells are reading accurately and showing changes during the cycle. During testing it was found that occasionally the load cells would take a tare value incorrect or wouldn’t output any signals. One possible cause of this was tension on the cables doing into the data logger. Another possible reason being due to the range in which the load cells are reading. They are rated to read up to 300kg and hence the drive cycle that was ran operates in the lower end of this. The next important value displayed is the current cycle time. This is so that the data produced by the data logger can be compared to the data output by the VESC tool connected to the UUT. In addition to this, the VESC would run a timed drive cycle for the UUT, therefor ensuring that the drive cycle times are coherent with what is happening. The final value displayed is the Torque, this is calculated live using both the load values. The program takes advantage of using loops and if statements to ensure that the data is recorded and displayed every 0.5 seconds.

```
while(Time - StartTime <= 60000){

}
```

Figure 36 - While Loop for a 60 Second Cycle Duration.

The actual refresh rate of the Arduino is much faster however due to the refresh rate of the TFT display, it is currently limited to 0.5 seconds.

4.2.3 Load, RPM and Torque Readings

For each load cell, every 0.5 seconds a live reading is taken. These raw values have their respective tare values deducted from them and then they are adjusted by the scale factors calculated during the calibration phase in the development. As both load cells are different, they require their own tare values and scale factors.

```
reading = ((scale.read()-tareLoad)/-1382.62)*0.93156;
reading2 = ((scale2.read()-tareLoad2)/-1552.61)*1.0433;
```

Figure 37 - Load Cell Readings with Calibration.

As the load range this investigation currently operates in is low, the load values are recorded as floats. This means that the Arduino stores them as decimal values. Currently set to 4 decimal places.

The RPM recording of the data logger happens constantly and the refresh rate of the Arduino, it is constantly looking for a dip in the value being outputted from the hall effect sensor. With no material in front of the hall effect (For this investigation and extended screw was used) the output value is 1 constantly. The if loop seen in figure 38 checks if the current sensor value is a LOW value (representing a drop in voltage) and the previous value is a HIGH. This is done to ensure that it only measures the presence of the bolt once per revolution of the UUT. When this loop triggers, it records the current time and subtracts the previous time, calculating the time difference. This time difference is then manipulated to create an RPM value.

```
sensorValue = digitalRead(TriggerPin);
Time = millis();
if (sensorValue == LOW && PrevSensorValue == HIGH) {
        timeDif = (Time-prevTimeHall);
        rpmValue = 60000/timeDif;
        prevTimeHall = Time;
}
PrevSensorValue = sensorValue;
```

Figure 38 - RPM Calculating IF Loop.

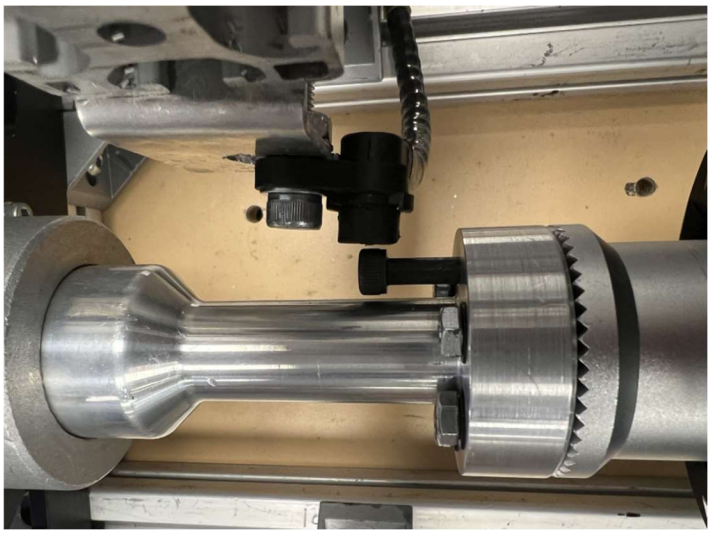

Figure 39 - Speed Sensor Set Up with Extended Bolt to Trigger a Reading.

With the current method of calculating RPM, the program sometimes misses the presence of the bolt on the dyno. This is due to the refresh rate at low RPMs. In order to improve this in the future, an interrupt will be added in place of this if statement to accurately record the value of RPM at all speeds.

The final data that is calculated is the torque being produced by the UUT. Using the values recorded from the load cells and a constant calculated by Alexandre About, a value for torque produced can easily be outputted. If the value of this constant needs to be adjusted or checked, the code is simple and easy to change allowing for future improvements to the Dyno.

```
TorqueReading = (absReading1+absReading2)*0.163;
```

Figure 40 - Torque Calculating Code.

4.2.4 Data Recording

The final main element of the coding is the data recording. During the initial experiments, the data was outputted into the Arduino IDE serial port and manually formatted to an excel file. However, an SD card library was added to the program allowing it to store the data directly, without the need for additional formatting.

This was a key objective of the dyno as without it, the dyno would require additional software and hardware, e.g. a laptop, reducing the overall accessibility of the rig. Without the implementation of the SD card, there were few solutions to exporting the data in a readable form. An example of an alternatives was to install 3rd party software's to record data values being received by the laptops port. This solution wasn't deemed feasible due to the need to install additional software to run the data logger.

```
//Save Data to SD Card
 myfile.print(TimeRec);
 myfile.print(" ");
 myfile.print(rpmValue);
 myfile.print(" ");
 myfile.print(reading);
 myfile.print(" ");
 myfile.print(reading2);
 myfile.print(" ");
 myfile.print(TorqueReading);
 myfile.println(" ");
```

Figure 41 - Program to Store Values onto SD Card.

Figure 41 shows the data that is being printed onto the SD card every 0.5 seconds during the cycle. In the future development of the dyno this data can be expanded to include other relevant values from the dyno, such as voltage/current of the resistive torque circuit, or the various temperatures of the rig.

4.3 Drive Cycle Results

For the final test, a duty cycle of increasing RPM was run with the resistive torque circuit producing a maximum load. Recording the RPM and Torque every 0.5 seconds for a minute the drive cycle was structures as so:

- 10 seconds 0 ERPM
- 10 seconds 2000 ERPM
- 10 seconds 4000 ERPM
- 10 seconds 6000 ERPM
- 10 seconds 8000 ERPM
- 10 seconds 0 ERPM

4.3.1 RPM vs ERPM

The VESC Tool requests and records speed using ERPM, ERPM is the electrical RPM which records the speed depending on the number of poles. The UUT used in this investigation has 14 pole pairs. Therefore, to get the actual RPM of the motor equation 3 is used.

$$RPM = ERPM/14 \qquad [3]$$

Plotting the RPM from the data logger against the ERPM shows the trend of data. As mentioned previously, at lower RPMs the data shows that logger misses the bolt, fluctuating the RPM but maintains a peak value at the expected speed. This can be improved further in the future by adding an additional bolt and doubling the time difference the data logger records between spikes in voltage.

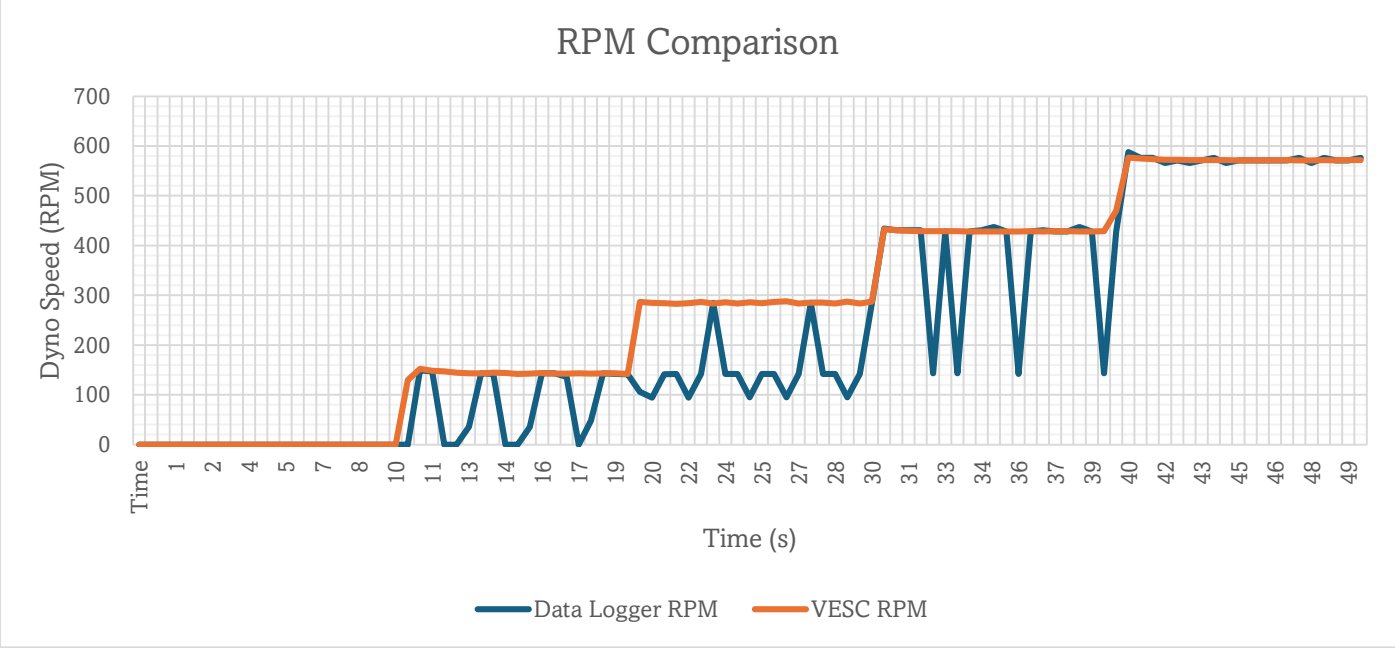

Figure 42 - Data logger RPM vs VESC RPM plot.

In the higher speed range, the data logger becomes more accurate, logging the actual speed value more frequently. At over 500 RPM the data logger becomes more accurate. At this high speed, the VESC had an average request RPM of 572.09 whereas the data logger recorded an average of 572.35 RPM. Using equation 4, a percentage of error can be calculated for the data logger. With the requested RPM from the VESC as the 'Accepted Value' and the average of the data loggers RPMs to be the 'Measured Value'.

$$Percent\ Error = \frac{Accepted\ Value - Measured\ Value}{Accepted\ Value}\ x\ 100 \qquad [4]$$

It is found that when requesting 572 RPM the data logger has a percentage of 0.05%. This shows that at high RPMs the data logger highly accurate in displaying the RPM.

4.3.2 Motor Efficiency

One way to compare the success of the dyno developed in this investigation, is by comparing the motors efficiency at various speeds. The formulas used to calculate motor efficiency is as follows:

$$Motor\ Efficiency = \frac{Mechanical\ Power}{Electrcial\ Power}\ x\ 100 \qquad [5]$$

$$Mechanical\ Power = Torque\ x\ Rotational\ Speed \qquad [6]$$

$$Electrical\ Power = Voltage\ x\ input\ current \qquad [7]$$

$$Rotation\ Speed\ (Radians\ per\ second) = RPM\ x\ \frac{2\pi}{60} \qquad [8]$$

Using the torque (Newton meters) calculated by the data logger, the rotation speed measured converted into radians per second, the voltage in and the input current recorded by the VESC tool, a motor efficiency of the UUT can be calculated.

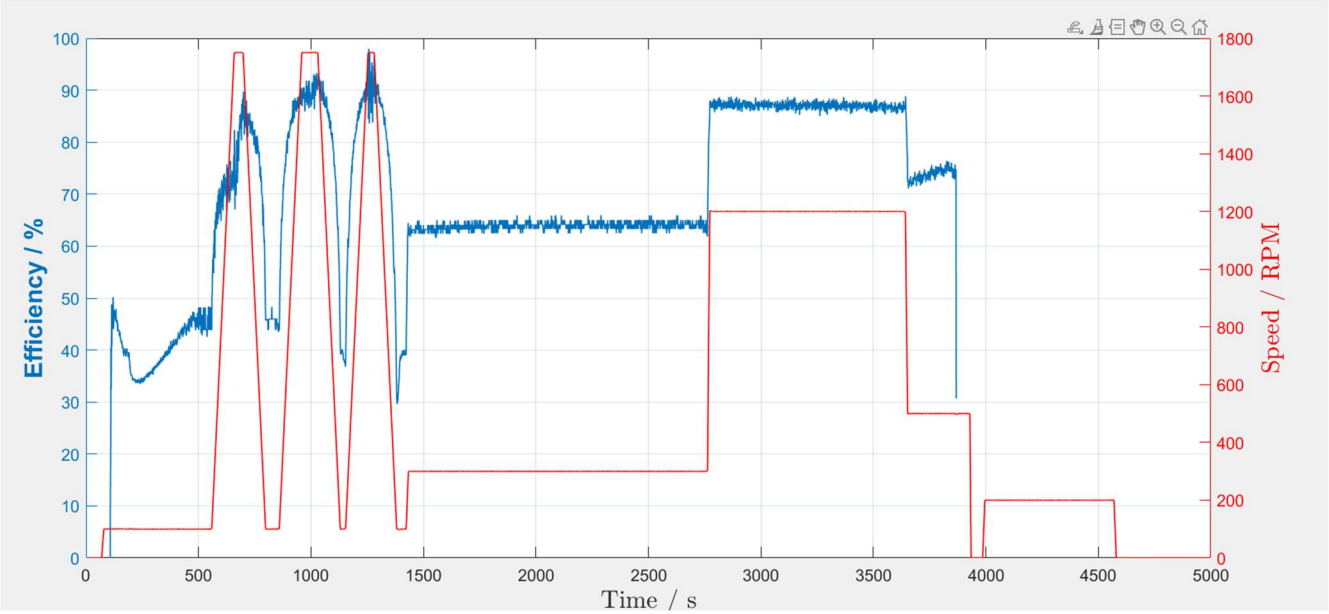

Figure 43 - Motor Efficiency from University Dyno Data.

Figure 42 is data provided by the university comparing motor efficiency at different speeds. It shows that at higher speeds, the motor is more efficient with efficiency at an average of 87% at 1200 RPM. Whereas at the lower speeds such as 300 RPM, the efficiency drops to an average of 64%. This decrease in efficiency could be due to the actual power being lower at low speeds, hence the fraction of power loss is a greater proportion of overall power.

A similar trend can be seen in the data obtained from the dyno developed in this investigation. Figure 43 displays the same variables to that recorded on the university's dyno. The data used to calculate the data displayed in the following figure was obtained from cycle 4 conducted on 29/04/24 with a load applied to the UUT. The data shows that at low speed the motor efficiency fluctuates significantly, this is due to an inaccuracy in the load cells and recorded torque. As the speed increases, at 428 RPM, the motor efficiency becomes steadier at an average of 82%.

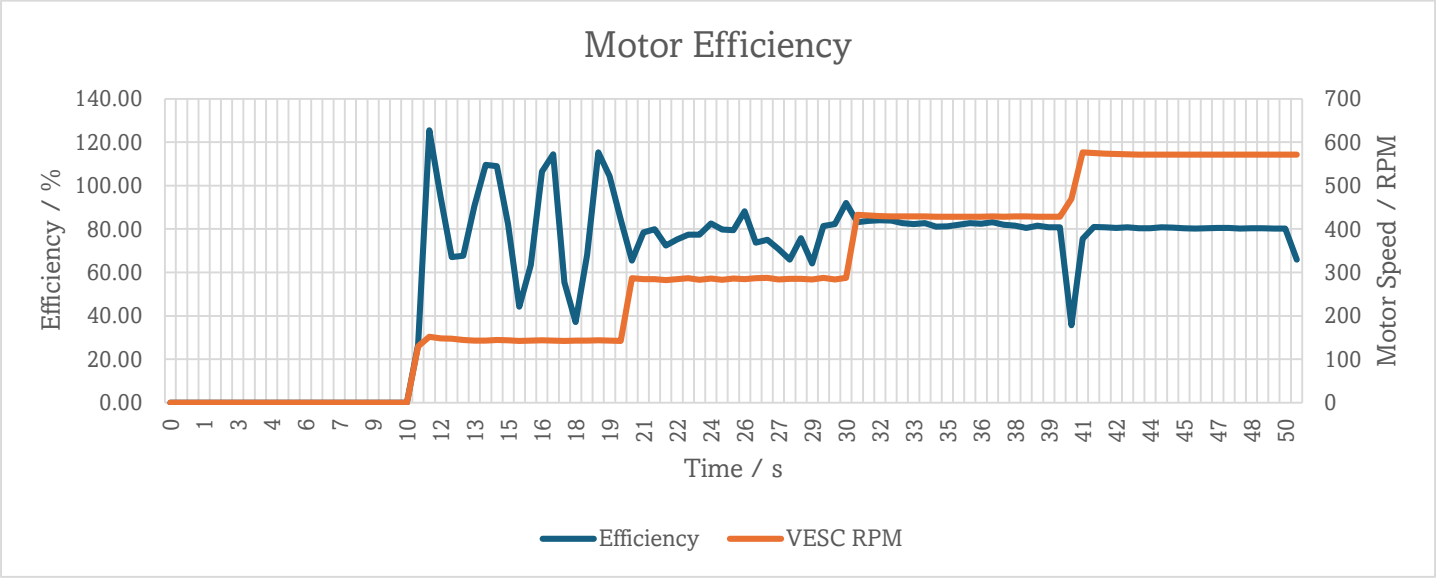

Figure 44 - Motor Efficiency recorded from this Investigation.

At 572 RPM, the TBRe Dyno recorded an average efficiency of 79%. Compared to the UoB Dyno which at 500RPM recorded an efficiency of 74%. This shows a correlation and similarities between the data obtained by the UoB dyno and the TBRe dyno.

4.3.3 UUT Torque

The final data required to evaluate the effectiveness of the data logger is the comparison of expected torque compared to calculated torque. The method for calculating the torque recorded from the data logger was explained previously in the software section of the results. For the ideal torque values, the motor efficiency is assumed to be 100%. There for the following equation is used.

$$Ideal\ Torque = \frac{Motor\ speed\ (rad/s)}{Electrical\ Power} \qquad [8]$$

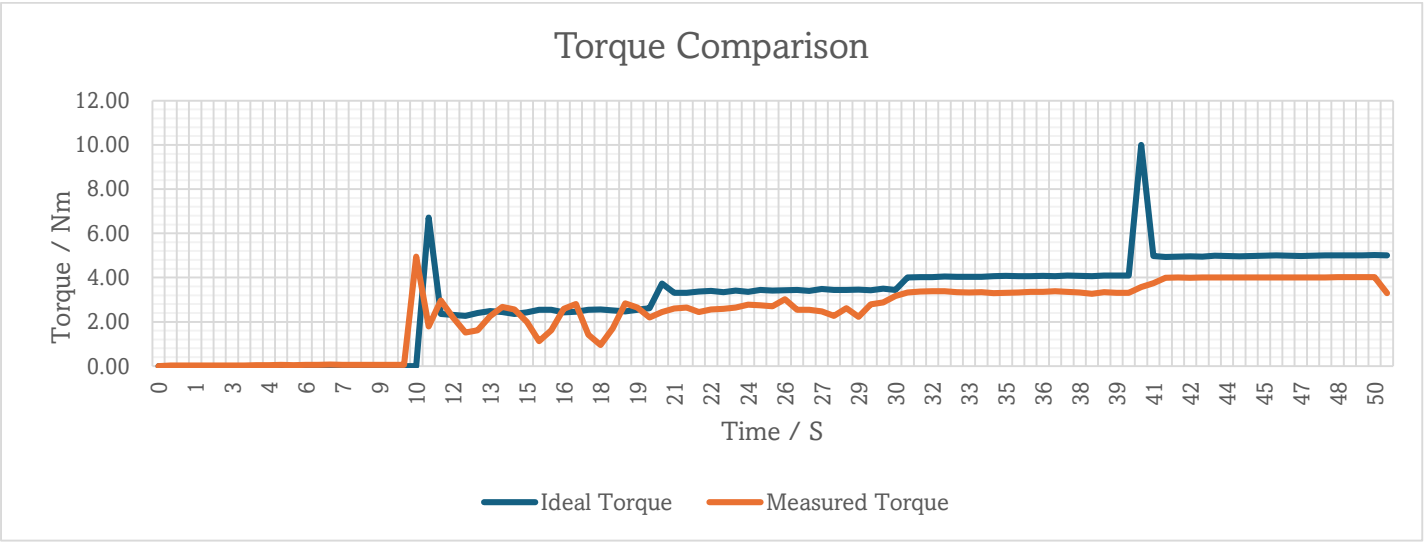

Figure 45 - Ideal Torque vs Measured Torque comparison.

Similar to RPM measured by the data logger, at lower speeds the motor is less accurate in recording a torque value and more accurate at higher speeds. At 572 RPM, the data logger recorded an average torque of 4.01 Nm where are the ideal value for torque at 100% motor efficiency should be 4.98 Nm. This presents a percentage error of 19.56%. However, assuming the motor is 79% efficient at around 500 RPM based of the UoB data. The expected torque should be around 3.69 Nm. Giving a percentage error of 8.71% which is more acceptable.

5 Discussion

Over all the data logging system developed for the TBRe Dynamometer successfully acquired and delivered relevant data relating to the unit under test. The effectiveness of the results can be split up into the key objectives of the project: Accuracy, Interface, and Accessibility.

5.1 Accuracy

From the results acquired from the drive cycle that was run, it can be determined that at speeds above 500 RPM, the data logger was accurate in displaying and reading live data from the load cells and hall effect speed sensor. Given a 0.05% error for the RPM and an 8.71% error for the Torque, the data logger was accurate enough to sufficiently provide data relating to the characteristic of the UUT. When it comes to the RPM accuracy, the test is assuming that the requested ERPM form the VESC tool is accurate to what is occurring on the dyno. In addition to this, another assumption that was made was that the motor is 100% efficient to produce the ideal torque.

5.1.1 Limitations and recommendations

Due to the short RPM range investigated in this report, there is little ability to predict how the data logger will perform at higher speeds. The main recommendation would be to test the data logger at higher speeds and for longer durations. The larger range of data would allow a more in-depth analysis into the accuracy of the data logger.

5.2 Interface

The data logger can efficiently display relevant data to the user during a drive cycle. This means that any individual operating the dyno should be able to use it with ease. As this was one of the key objectives for this investigation, it is important that the data produced is live and accurate. Due to time constraints, the analysis of the layout on the interface could not be conducted. To further evaluate the success of the layout, it would require additional studies into what data TBRe require to be displayed and what can just be recorded.

5.1.1 Limitations and recommendations

A key limitation when it comes to the interface of the data logger is the refresh rate of the screen. A recommendation to overcome this would be to implement an additional Arduino nano which is programmed to record and store data. This allows the program to be shorter as it is split, creating a faster reading Arduino and data logger.

5.3 Accessibility and Design

The final design of the data logger is an easy to operate tool, allowing for quick and accurate reading and recording of data. The design is clear and simple, reducing the chance for human error when operating. However, with no quantitative data to evaluate the effectiveness of the design it is hard to represent the results. A future recommendation would be to investigate the qualitive features of the design. One key area of consideration would be the ergonomics of the data logger. This could be done via surveys and focus groups.

5.2 Other Inaccuracies

5.2.1 Calibrations in Load Cells

One major inaccuracy is in the calibration of the load cells and the calculating of the scale factor. In order to accurately calibrate the load cells, a larger known mass would need to be used. During this investigation, a mass of 1.25kg was used during the calibration process. A future recommendation would be to calibrate them with a mass of 100kg. As the load cells are rated up to 300kg, working in the lowest end of the that range leads to a large uncertainty.

5.2.2 General Dyno Structure

A possible cause of inaccuracy could be due to the containment of the dynamometer. Traditionally dynos are heavy pieces of machinery, often fixed to the ground. The TBRe Dyno sits on a table which is free to move, this leads to vibration and displacement of the rig during RPM changes. These movements in the rig will lead to inaccuracies in the values obtained by the load cells. A simple improvement to the dyno that would be favourable is to secure it to the ground. This significantly reduces the ability for the dyno to move and vibrate. Alternatively, rubber mounts could be used or other anti-vibration materials.

6 Conclusions

Overall, the data logger has successfully achieved its objectives in accurately recording and displaying live data from the dynamometer. Obtaining a percentage error of only 0.05% in high RPM readings and 8.71% error in torque reading. Whilst there is room for improvement when it comes to the lower ranges of RPM, this may just be a limitation of the dyno. Given upon beginning this project, I had little knowledge of Arduino programming and electrical circuitry, the development of the data logger has been successful. Learning both MATLAB and ARDUINO IDE in order to develop the dynamometer. The software has been designed so that it can be easily improved upon should there be the need to. The hardware of the data logger is simple and effective, it is easy and safe to use whilst reducing any inaccuracies in the circuitry.

Whilst it was effective during the drive cycles conducted, further testing is required to confirm the data logger's effectiveness for all UUTs. Given the current low range of testing, the ability to predict whether the calibration of the load cells is accurate in addition to whether the Arduino has the ability to measure motor speeds up to 7000RPM which is outlined as a key objective.

7 Future work

7.1 Further Testing

The further development of the dyno would require additional testing. Due to time constraints and limited resources, the tests conducted in this investigation were limited. Ideally, more repeats would be taken and ranging across larger RPM values. This would help rule out any inaccuracies and anomalous results that may have occurred during this investigation.

In order to test at higher speeds, a larger battery would need to be sourced. The investigation is currently limited due to the max power produced by the batteries being used. With future development and investment into the dynamometer, a higher power rated battery can be obtained increasing the capabilities of motor testing. This would help achieve the objective outlines in the beginning of this report to achieve RPM values up to 7000. This value is the maximum value at which a motor would currently be expected to spin at if it were attached to the competition car.

Another benefit to operating at the high RPM ranges is that it would allow the load cells to read data across the entire range. With their current load rating up to a maximum of 300kg (2942 Newtons), the current readings from the dyno measuring at a maximum of 14 Newtons. This means, this experiment is currently recording data in the bottom 0.47% of the load cells abilities.

7.2 Software Improvements

7.2.1 UUT Control

The main improvement to the software of the data logger in the long run would be to have the ability to control the Unit under test with a preloaded/pre-set drive cycle. The Arduino should have the ability to control the UUT via a PWM output. This would allow the removal of the final laptop to run the dyno. Creating an additional menu screen on the display to configure the drive cycle, the dyno would become its own entity. An alternative way that the drive cycle could be uploaded is via the SD card, allowing for more customisation by the user needing to test a motor.

7.2.2 RPM Accuracy

As mentioned previously in the report, one future improvement to the software would be to alter the way it records RPM. By swapping the current IF statement with an interrupt, the Arduino will be able to read the RPM at a higher accuracy at the lower speed range. In addition to this, it will allow for more frequent data reading. As with the current program, when attempting the increase, the frequency at while the data is recorded, the RPM stops working.

7.2.3 Aesthetic improvements

The current display of the data logger is very simplistic and plain, with more time this can be changed. Displaying the live readings in a better format and style. One change that will be made is the orientation of the display. Allowing for more space to display larger values as with the current RPM running low, at higher speeds the values may no longer be visible.

7.3 Dyno Improvements

7.3.1 Additional Motor

One improvement to the overall dyno what will benefit the data logging is the swapping of the DC Motor circuit with a spare motor the university have access to. With the current set up, the resistive torque can only be varied between 3 values. With the addition of the motor in figure 42, The individual using the dyno would be able to request a specific resistive torque and vary it during a drive cycle. This allows for further analysis of the UUT and how it reacts to different loads and varying speeds.

Figure 46 - 30KW Motor by FREERCHOBBY. (FREERCHOBBY, n.d.)

7.3.2 Driver in the loop

A possible end goal for the dyno would be the possibility of linking the UUT to a simulation and having direct feedback from a driver inputting the power request to the UUT. This would allow the TBRe Team to fully understand if the motor is suitable for the vehicle and how it will react in a more dynamic environment.

8 References

Amazon, n.d. [Online]
Available at: https://www.amazon.co.uk/3-5inch-480x320-Resolution-Support-Arduino/dp/B07N4KXP5X?th=1
[Accessed 20 4 24].

Archive/uig, U. H., 2016. *Dynamometer.* [Art].

Arduino, n.d. *Adafruit GFX Library.* [Online]
Available at: https://www.arduino.cc/reference/en/libraries/adafruit-gfx-library/
[Accessed 5 5 24].

Arduino, n.d. *Adafruit TouchScreen.* [Online]
Available at: https://www.arduino.cc/reference/en/libraries/adafruit-touchscreen/
[Accessed 5 5 24].

Arduino, n.d. *Arduino Mega 2560 Rev3.* [Online]
Available at: https://store.arduino.cc/products/arduino-mega-2560-rev3
[Accessed 02 05 2024].

Arduino, n.d. *https://www.arduino.cc/reference/en/libraries/mcufriend_kbv/.* [Online]
Available at: https://www.arduino.cc/reference/en/libraries/mcufriend_kbv/
[Accessed 5 5 24].

Arduino, n.d. *HX711 Arduino Library.* [Online]
Available at: https://www.arduino.cc/reference/en/libraries/hx711-arduino-library/
[Accessed 5 5 24].

Arduino, n.d. *SD.* [Online]
Available at: https://www.arduino.cc/reference/en/libraries/sd/
[Accessed 5 5 24].

BOSCH, n.d. *Speed Sensor Hall-Effect Mini-HA-P.* [Online]
Available at: https://www.bosch-motorsport.com/content/downloads/Raceparts/en-GB/53304715208315403.html#/Tabs=53320587/
[Accessed 20 04 2024].

Electric Motorsport, 2016. *Motenergy ME1616 Brushless 20 kW - 55 kW Liquid-Cooled IPM Motor 48-120V.* [Online]
Available at: https://www.electricmotorsport.com/me1616-brushless-65hp-liquid-cooled-ipm-motor-24-120v.html
[Accessed 30 04 2024].

FREERCHOBBY, n.d. *FRC 25KW MP120100 KV50 Outrunner Brushless Motor for Electric Paramotors and Electric Go-karts.* [Online]
Available at: https://www.freerchobby.cc/products/frc-25kw-mp120100-kv50-outrunner-brushless-motor-for-electric-paramotors-and-electric-go-karts?variant=34581817589921
[Accessed 27 03 2024].

FREERCHOBBY, n.d. *MP 15470 30kw 60kg Thrust Sensorless Motor for Drone /Paramotor.* [Online]
Available at: https://www.freerchobby.cc/products/mp-15470-30kw-60kg-thrust-sensorless-motor-for-drone-paramotor
[Accessed 5 5 24].

Horne , D. & Talbot, E., n.d. *The History of the Régnier Dynamometer.* [Online]
Available at: https://www.gilai.com/article_31/The-History-of-the-Regnier-Dynamometer
[Accessed 20 04 2024].

Industrial Quick Search, n.d. *Dynamometers.* [Online]
Available at: https://www.iqsdirectory.com/articles/dynamometers.html
[Accessed 01 05 2024].

Institution of Mechanical Engineers, 2024. *FORMULA STUDENT.* [Online]
Available at: https://www.imeche.org/events/formula-student/about-formula-student
[Accessed 29 04 2024].

Loude, J.-F., 2014. Early Dynamometers (from Muscle to Steam Power). *XXXIII Scientific Instrument Symposium,* 27 August .

Mucevski, K., 2015. *Inductive and Hall Effect RPM Sensors Explained*. [Online]
Available at: https://www.linkedin.com/pulse/inductive-hall-effect-rpm-sensors-explained-kiril-mucevski/
[Accessed 02 03 2024].

Pushton Electronic Equipment, n.d. *PUSHTON*. [Online]
Available at: http://www.pushton.com/?m=home&c=View&a=index&aid=122
[Accessed 24 1 24].

Quartz Components, n.d. *HX711 Load Cell Amplifier Module*. [Online]
Available at: https://quartzcomponents.com/products/hx711-load-cell-amplifier-module
[Accessed 1 5 2024].

Romandoni, N., 2021. Design of Water Brake Dynamometer. *Journal of Physics: Conference Series,* Issue 1845.

RS, n.d. *A Complete Guide to Load Cells*. [Online]
Available at: https://uk.rs-online.com/web/content/discovery/ideas-and-advice/load-cells-guide
[Accessed 20 4 24].

RS, n.d. *The Guide to Hall Effect Sensors*. [Online]
Available at: https://uk.rs-online.com/web/content/discovery/ideas-and-advice/hall-effect-sensors-guide
[Accessed 2 5 25].

Ryan, 2021. *What Are Some Pros and Cons of TFT Displays?*. [Online]
Available at: https://nauticomp.com/what-are-some-pros-and-cons-of-tft-displays/
[Accessed 01 05 24].

Team Bath Racing electric, n.d. *TBRe Home*. [Online]
Available at: https://www.teambathracingelectric.com/
[Accessed 25 04 2024].

TRAMPA BOARDS, n.d. *VESC 100/250*. [Online]
Available at: https://trampaboards.com/vesc-100250--p-27368.html
[Accessed 02 05 2024].

Yida, n.d. *10 Things you can do with your HX711 and Load Cell*. [Online]
Available at: https://www.seeedstudio.com/blog/2019/11/26/10-things-you-can-do-with-your-hx711-and-load-cell/#:~:text=What%20is%20the%20HX711%3F&text=The%20HX711%20is%20a%20precision,reporting%20them%20to%20another%20microcontroller.
[Accessed 24 04 2024].

9 Appendices

Appendix A – Matlab Program to display and record load cell readings; [Generated using Chat GPT then modified to users' needs]

```
clear
a = arduino ('COM5','mega2560','libraries','basicHX711/basic_HX711');
LoadCell=addon(a,'basicHX711/basic_HX711',{'D25','D27'});

%Take TareWeight Measurement
tareWeightLoop = 1
while tareWeightLoop < 20
    tareWeight(tareWeightLoop) = read_HX711(LoadCell);
    tareWeightLoop = tareWeightLoop + 1;
end
AvgTareWeight = mean(tareWeight)

k = 1 ;
NoRuns = 100;
PauseTime = 0.5
LoadCycle = zeros(NoRuns,2);
LoopTimes = NoRuns+1;

while k < 101
    scaled = (read_HX711(LoadCell))-AvgTareWeight;
    weight = (scaled/-14772.72);
    LoadCycle(k,1) = [weight];
    LoadCycle(k,2) = [k];
    pause(PauseTime);
    k = k + 1
end


colNames = {'Weight (KG)','Reading'};
LoadCycleTable = array2table(LoadCycle,'VariableNames',colNames)
%xlswrite('LoadCycle.xlsx',LoadCycleTable);
plot(LoadCycleTable,'Reading','Weight (KG)')
%print('Load Cycle end')
```

Appendix B – Matlab program to display RPM ; [Generated using Chat GPT then modified to users' needs]

```
%% Define Arduino object
clear
arduinoObj = arduino ('com3','mega2560','libraries','basicHX711/basic_HX711'); % Change 'COM3'
to your Arduino port

% Define the analog pin where the Hall effect sensor is connected
analogPin = 'A0';
configurePin(arduinoObj,analogPin,'Analoginput');
% Set up a live plot
figure;
hLine = animatedline;
xlabel('Time');
ylabel('Hall Effect Sensor Reading');

% Set up a loop to continuously read and plot sensor data
try
    while true
        % Read the sensor value
        sensorValue = readVoltage(arduinoObj, analogPin);

        % Get current time for x-axis
        t = datetime('now');
```

```
        % Add new data point to the plot
        addpoints(hLine, datenum(t), sensorValue);
        datetick('x', 'keeplimits');
        drawnow;
    end
catch
    disp('Error reading from Arduino. Closing plot...');
    clear arduinoObj;
    close;
end
```

Appendix C – Arduino IDE code for the data logger ; [Base Touch Screen code developed by Alex Cucchiara then Modified and Developed]

```
//Declare Variables
long prevTimeHall;
long timeDif;
long Time = 0;
float TimeRec = 0;

//long ThreshHold = (-1/5)*1023;
long StartTime;
long Time2 = 0;
long prevTime2;
int val;
int prevVal;
//Define the Pins
#define TriggerPin 19
const int LOADCELL_DOUT_PIN = 27;
const int LOADCELL_SCK_PIN = 25;
const int LOADCELL_DOUT_PIN_2 = 51;
const int LOADCELL_SCK_PIN_2 = 49;

//Variables for RPM and Load Cell Reading/formatting
int rpmValue;
float reading;
float reading2;
float absReading1;
float absReading2;
float TorqueReading;
float tareLoad;
float tareLoad2;
bool sensorValue;
bool PrevSensorValue = true;


//Include the relevant adafruit and touch screen libraries
#include <Adafruit_GFX.h>
#include <MCUFRIEND_kbv.h>
MCUFRIEND_kbv tft;
#include <TouchScreen.h>
```

```
#include <Adafruit_I2CDevice.h>
#include <HX711.h>
#include <SPI.h>
#include <SD.h>

//Define the min and max pressure bounds for the touch screen,
// any press between these values is counted as a touch
#define MINPRESSURE 100
#define MAXPRESSURE 1000

//define two variables that track the current and previous value of the data


//Instantiate scale object
HX711 scale;
HX711 scale2;

//SD setup
File myfile;
#define CS_Pin 4

// ALL Touch panels and wiring is DIFFERENT
// copy-paste results from TouchScreen_Calibr_native.ino

//TouchScreen Calibration (Seperate Script to obtain values)
const int XP=8,XM=A2,YP=A3,YM=9; //320x480 ID=0x9486
const int TS_LEFT=148,TS_RT=867,TS_TOP=944,TS_BOT=94;

//create an instance of the touchscreen
TouchScreen ts = TouchScreen(XP, YP, XM, YM, 300);

//define the two buttons as on button
Adafruit_GFX_Button on_btn;


//This function stores the x,y location of a press on the screen
//only if it is between the pressure limits
int pixel_x;
int pixel_y;     //Touch_getXY() updates global vars
bool Touch_getXY(void)
{
    TSPoint p = ts.getPoint();
    pinMode(YP, OUTPUT);      //restore shared pins
    pinMode(XM, OUTPUT);
    digitalWrite(YP, HIGH);   //because TFT control pins
    digitalWrite(XM, HIGH);
    bool pressed = (p.z > MINPRESSURE && p.z < MAXPRESSURE);
    if (pressed) {
      //maps the pixel values according the the screen calibration values
        pixel_x = map(p.x, TS_LEFT, TS_RT, 0, tft.width());
        pixel_y = map(p.y, TS_TOP, TS_BOT, 0, tft.height());
    }
```

```
    return pressed;
}
//define the colours that can be used on the screen
#define BLACK   0x0000
#define BLUE    0x001F
#define RED     0xF800
#define GREEN   0x07E0
#define CYAN    0x07FF
#define MAGENTA 0xF81F
#define YELLOW  0xFFE0
#define WHITE   0xFFFF

void setup(void)
{
  //start up the serial monitor
    Serial.begin(9600);
  //Assign Load Cell Pins and Hall Effect
    scale.begin(LOADCELL_DOUT_PIN, LOADCELL_SCK_PIN);
    scale2.begin(LOADCELL_DOUT_PIN_2, LOADCELL_SCK_PIN_2);
    pinMode(TriggerPin, INPUT_PULLUP);
    //attachInterrupt(digitalPinToInterrupt(TriggerPin),  rpmISR, CHANGE);
    //screen start up code
    uint16_t ID = tft.readID();
    if (ID == 0xD3D3) ID = 0x9486; // write-only shield
    tft.begin(ID);
    tft.setRotation(0);
    //white background
    tft.fillScreen(WHITE);

    //define where the two buttons will go on the screen
    on_btn.initButton(&tft,  150, 50, 150, 50, WHITE, GREEN, BLACK, "BEGIN", 2);
    on_btn.drawButton(false);
}



void loop(void)
{
  //call screen pressed function
  bool down = Touch_getXY();

  //check if the press is on either of the buttons
  on_btn.press(down && on_btn.contains(pixel_x, pixel_y));
  if (on_btn.justReleased())
      on_btn.drawButton(false);
  //code to execute when button pressed
  if (on_btn.justPressed()) {
    tft.fillRect(51, 100, 260, 329, WHITE);
      on_btn.drawButton(true);

      //Measure the tare values of the load cells
      tareLoad = scale.read();
```

```
      tareLoad2 = scale2.read();

      //Start the cycle timer
      StartTime = millis();

      //Starts a 1 minute long loop
      while(Time - StartTime <= 60000){

      //Measures the digital value from the halleffect sensor and records the current time
      sensorValue = digitalRead(TriggerPin);
      Time = millis();

      //Checks for a dip in the hall effect reading and records the time difference between
that and the last drop
       if (sensorValue == LOW && PrevSensorValue == HIGH) {
         timeDif = (Time-prevTimeHall);
         rpmValue = 60000/timeDif;
         prevTimeHall = Time;
       }
       PrevSensorValue = sensorValue;
      Time2 = millis();
      if (Time2-prevTime2 >= 500){

        tft.setTextColor(BLACK);
        tft.fillRect(149,149,150,50,WHITE);
        tft.fillRect(49,299,400,50,WHITE);
        tft.fillRect(125,450,100,50,WHITE);
        tft.fillRect(200, 350,100,50,WHITE);


        prevTime2 = Time2;
        //Display RPM value
        printmsg(150,100,"RPM",2);
        dispValue(150,150,4,rpmValue);

        //reads and displays Load 1 in Newtons
        reading = ((scale.read()-tareLoad)/-1382.62)*0.93156;
        printmsg(50,250,"LOAD 1",2);
        tft.setCursor(50, 300);
        tft.println(reading, 4);

        reading2 = ((scale2.read()-tareLoad2)/-1552.61)*1.0433;
        printmsg(200,250,"LOAD 2",2);
        tft.setCursor(200, 300);
        tft.println(reading2, 4);

        absReading1 = fabsf(reading);
        absReading2 = fabsf(reading2);
        TorqueReading = (absReading1+absReading2)*0.163;
        printmsg(125,400,"TORQUE",2);
        tft.setCursor(125, 450);
        tft.println(TorqueReading, 4);
```

```
        delay(5);
        TimeRec = Time-StartTime;

        printmsg(100,350,"TIME : ",2);
        tft.setCursor(200, 350);
        tft.println(TimeRec/1000,2);

        //Save Data to SD Card
        myfile.print(TimeRec);
        myfile.print(" ");
        myfile.print(rpmValue);
        myfile.print(" ");
        myfile.print(reading);
        myfile.print(" ");
        myfile.print(reading2);
        myfile.print(" ");
        myfile.print(TorqueReading);
        myfile.println(" ");
        rpmValue = 0;

      }
    }
  }
}

void dispValue(int x, int y, int sz, long msg)
{
    tft.setCursor(x, y);
    tft.setTextSize(sz);
    tft.println(msg);
}
void printmsg(int col, int row, const char *msg, int sz)
{
    //tft.setTextColor(YELLOW, BLACK);
    tft.setTextSize(sz);
    tft.setCursor(col, row);
    tft.println(msg);

}
```

Appendix D – MATLAB code used to display UoB Dyno Data; {Code obtained from Alexandre About and further modified}

```
% Script used to select specific files to plot data, according to a specific file structure.

clc
clear all
hold off
%% Load the Excel data
% Prompt the user to select a file
[filename, filepath] = uigetfile('*.*', 'Select a file to read');
% Check if the user canceled the operation
if isequal(filename,0)
    disp('User canceled the operation. Exiting script.');
    return;
end
% Construct the full file path
fullpath = fullfile(filepath, filename);
% Load the selected file
data = readtable(fullpath);
% Display a message indicating successful file loading
disp(['File "', filename, '" loaded successfully.']);
%% Compute Power and efficiency
%number of rows
N = size(data.Time,1);
% Define power
power_in = [];
power_out = [];
% computing power
for n = 1:N
    power_in(:,n) = data.input_current(n)*data.Status_InputVoltage(n);
    power_out(:,n) = data.motor_torque(n)*data.motor_speed(n)*(2*pi/60);
end
%compute efficiency
efficiency = power_out./power_in;
%%Plotting
figure(1)
clf;
plot_template_2axes(data.Time, data.Status_InputVoltage, data.input_current, ...
    'Time / s', 'Voltage', 'Current');
hold on
figure(2)
clf;
plot_template_2axes(data.Time, data.motor_speed, data.motor_torque,  ...
    'Time / s',  'Speed / RPM', 'Torque / Nm');
hold on
figure(3)
clf;
plot_template(data.Time, power_in, 'Time / s', 'Power / W');
hold on
plot(data.Time, power_out);
legend('Power In', 'Power Out')
figure(4)
clf;
% plot_template(data.Time, efficiency, 'Time (s)', 'Efficiency');
efficiency = efficiency*100;
plot_template_2axes(data.Time, efficiency, data.motor_speed,'Time / s', 'Efficiency / %',
'Speed / RPM');
ylim([0 100])
hold on
figure(5)
clf;
plot_template(data.motor_speed, data.motor_torque, 'Speed / RPM', 'Torque / Nm')
hold on
```

```
has context menu
```

Appendix E – MATLAB function used to display UoB Dyno Data; [Function obtained by Nathan Sell and edited by Alexandre About}

```
function plot_template_2axes(XData, YData1, YData2, Xtitle, Ytitle1, Ytitle2, varargin)
% PLOT_TEMPLATE Function which produces plots with a defined template
% Function created to simplify plot-making and shorten main script.
% Function specifically plots data with two separate sets and vertical axes
hold off
yyaxis left
plot(XData, YData1)
hold on
yyaxis right
plot(XData, YData2, 'r')
% Make all lines 2pt
set(findall(gca, 'Type', 'Line'),'LineWidth',1);
% Set axis labels to 12pt
set(gca,'fontsize',12)
yyaxis right
set(gca, 'ycolor', 'r');
% Add labels
xlabel(Xtitle,'fontsize',18,'fontweight','bold','interpreter','latex');
yyaxis left
ylabel(Ytitle1,'fontsize',18,'fontweight','bold','interpreter','latex');
yyaxis right
ylabel(Ytitle2,'fontsize',18,'fontweight','bold','interpreter','latex');
yyaxis left % switch back to left axis
grid on
if nargin==7
    legend(cellstr(varargin{1}),'fontsize',12,'interpreter','latex','location','NE');
    legend boxoff
elseif nargin==8
    legend(cellstr(varargin{1}),'fontsize',12,'interpreter','latex','location',varargin{2});
    legend boxoff
end
end
```

Appendix F – MATLAB function used to display UoB Dyno Data; [Function obtained by Nathan Sell and edited by Alexandre About}

```
function plot_template(XData, YData, Xtitle, Ytitle, varargin)
% PLOT_TEMPLATE Function which produces plots with a defined template
% Function created to simplify plot-making and shorten main script.
% Edited from function created by Nathan Sell.
% ALEXANDRE ABOUT - UNI OF BATH
hold off
plot(XData, YData)
%make all lines 2pt
set(findall(gca, 'Type', 'Line'),'LineWidth',2);
%set axis labels to 12pt
set(gca,'fontsize',12)
% Add labels
xlabel(Xtitle,'fontsize',18,'fontweight','bold','interpreter','latex');
ylabel(Ytitle,'fontsize',18,'fontweight','bold','interpreter','latex');
grid on
if nargin==5
    legend(cellstr(varargin{1}),'fontsize',12,'interpreter','latex','location','NE');
    legend boxoff
elseif nargin==6
    legend(cellstr(varargin{1}),'fontsize',12,'interpreter','latex','location',varargin{2});
    legend boxoff
end
end
has context menu
```

Appendix G – TBRe Dyno data obtained from VESC tool {VESC data acquired from Alexandre About}

Time	Power	Current in	Current motor	Temp in	ERPM	Voltage	Duty	Column 10	Column 11	Column 12
0	0	0	0	27.5	0	24.6	-0.3	24.6	21.6	21.4
0	0	0	0	27.5	0	24.6	-0.19	24.678	21.6	21.4
1	0	0	0	27.5	0	24.6	-0.02	24.6	21.5	21.3
1	0	0	0	27.5	0	24.6	-0.13	24.6	21.6	21.4
2	0	0	0	27.5	0	24.6	-0.564	24.688	21.6	21.4
2	0	0	0	27.5	0	24.6	-0.4	24.6	21.51	21.4
3	0	0	0	27.5	0	24.6	-0.208	24.6	21.6	21.4
3	0	0	0	27.5	0	24.6	0.246	24.6	21.6	21.4
4	0	0	0	27.5	0	24.6	0.132	24.6	21.5	21.3
4	0	0	0	27.5	0	24.6	-0.3	24.598	21.698	21.4
5	0	0	0	27.5	0	24.6	-0.22	24.44	21.44	21.24
5	0	0	0	27.5	0	24.6	-0.342	24.6	21.5	21.3
6	0	0	0	27.5	0	24.6	0.056	24.6	21.6	21.4
6	0	0	0	27.5	0	24.6	-0.3	24.6	21.586	21.386
7	0	0	0	27.5	0	24.6	-0.3	24.6	21.5	21.3
7	0	0	0	27.5	0	24.6	-0.33	24.6	21.6	21.4
8	0	0	0	27.5	0	24.6	-0.352	24.6	21.6	21.4
8	0	0	0	27.5	0	24.6	-0.3	24.674	21.526	21.326
9	0	0	0	27.5	0	24.6	-0.2	24.6	21.5	21.3
9	0	0	0	27.5	0	24.6	0.2	24.6	21.6	21.4
10	0	0	0	27.5	0	24.6	-0.32	24.6	21.56	21.36
10	0	0	0	27.5	0	24.6	0.072	24.6	21.562	21.4
11	98.59696	3.6988	17.5776	26.704	1814.08	24.6	26.188	24.6	21.584	21.384
11	41.0166	1.5188	8.1746	27.006	2128.04	24.7	21.942	24.706	21.506	21.4
12	39.22272	1.4488	7.9816	27.072	2075.2	24.828	22.212	24.9	21.6	21.4
12	37.5255	1.395	7.78	26.9	2060	25.05	21.8	25.05	21.6	21.4
13	38.51004	1.4316	8.0664	26.9	2020.84	25.3	22.316	25.3	21.528	21.328
13	39.25036	1.4588	8.2352	26.906	2003.82	25.5	22.006	25.5	21.506	21.4
14	38.37272	1.4204	8.0156	27.016	2003.76	25.7	21.9	25.616	21.6	21.4
14	36.99	1.37	7.769	27	2019.24	25.9	20.638	25.9	21.538	21.4
15	37.7694	1.402	7.956	26.94	2009.6	26.06	21.22	26	21.6	21.46
15	38.8314	1.4382	8.1364	27	1985.9	26.2	21.754	26.2	21.6	21.418
16	38.8424	1.4384	8.1116	27.004	1999.2	26.396	20.904	26.3	21.596	21.396
16	37.1898	1.3774	7.7666	27	2012.82	26.5	21.37	26.5	21.6	21.4
17	37.32044	1.3796	7.7972	27.052	1998.68	26.6	21.452	26.6	21.648	21.448
17	38.5091	1.421	8.08	27.1	1992.6	26.7	21.56	26.7	21.6	21.43
18	38.61	1.43	8.1276	27	2001.84	26.8	21.132	26.8	21.6	21.4
18	37.6866	1.3958	7.9206	27	1997.96	26.9	21.944	26.9	21.686	21.486
19	37.127	1.37	7.7244	27.1	2006.76	27	21.14	27	21.664	21.464
19	37.8864	1.4032	7.9386	27	2001.78	27.1	20.884	27.1	21.6	21.5
20	38.556	1.428	8.108	27	1990	27.2	21.32	27.1	21.68	21.48
20	109.47982	4.1156	11.2456	26.602	4017.54	27.2	41.582	27.202	21.6	21.4
21	95.19548	3.6196	13.094	26.3	3986.16	27.3	36.732	27.3	21.576	21.376
21	95.24692	3.6016	12.917	26.446	3978.72	27.4	39.22	27.346	21.7	21.5
22	96.1376	3.6372	13.0132	26.432	3956.28	27.4	41.804	27.4	21.6	21.432
22	96.9418	3.686	13.315	26.3	3981.7	27.5	41.3	27.5	21.69	21.49
23	96.73688	3.6488	13.112	26.512	4010.52	27.5	36.92	27.676	21.612	21.412
24	96.95994	3.6636	13.194	26.466	3963.98	27.6	42.434	27.6	21.634	21.434
24	96.33356	3.6276	13.0812	26.556	4001.96	27.7	37.78	27.6	21.712	21.456

25	97.26398	3.6812	13.3892	26.422	3966.76	27.7	42.504	27.7	21.7	21.5
25	96.784	3.68	13.13	26.3	4008	27.8	41.5	27.7	21.7	21.4
26	97.1331	3.6654	13.3872	26.5	3980.48	27.8	40.318	27.778	21.7	21.5
26	97.70992	3.7072	13.3796	26.356	4015.84	27.9	38.812	27.856	21.744	21.5
27	97.20808	3.6728	13.4764	26.468	4029.2	27.9	35.978	27.9	21.734	21.534
27	98.43312	3.7128	13.3608	26.512	3970.12	27.9	40.936	27.988	21.788	21.588
28	96.5999	3.673	13.303	26.3	3993.5	28	41.77	28	21.79	21.59
28	97.52	3.68	13.294	26.5	3996.36	28	38.844	28	21.768	21.568
29	96.45564	3.66	13.2736	26.354	3967.84	28.054	41.992	28.054	21.754	21.554
29	97.00368	3.6672	13.4712	26.452	4022.28	28.1	36.584	28.1	21.724	21.524
30	98.0341	3.6994	13.4644	26.5	3969.04	28.1	42.284	28.198	21.702	21.498
30	97.3838	3.686	13.238	26.42	4020.6	28.1	37.14	28.1	21.8	21.6
31	164.26248	6.4272	12.515	25.558	6056.12	28.2	60.8	28.2	21.8	21.6
31	164.54036	6.4364	12.512	25.564	6033.24	28.2	60.3	28.2	21.728	21.528
32	164.38784	6.4214	12.5114	25.6	6017.56	28.2	59.958	28.286	21.814	21.614
32	164.30064	6.42	12.5132	25.592	6008.08	28.3	60.156	28.208	21.808	21.6
33	163.608	6.416	12.533	25.5	6007	28.3	60.19	28.3	21.8	21.6
33	163.0254	6.4052	12.5112	25.452	6005.4	28.3	60.676	28.3	21.852	21.552
34	162.69208	6.4052	12.5026	25.4	6004.22	28.3	60.104	28.3	21.874	21.6
34	163.29152	6.4288	12.5476	25.4	5997.08	28.4	60.204	28.4	21.804	21.6
35	163.32346	6.4346	12.5464	25.382	5999.54	28.4	59.69	28.318	21.8	21.6
35	162.9664	6.416	12.528	25.4	5998.6	28.4	60.24	28.36	21.86	21.62
36	163.068	6.42	12.5438	25.4	5997.86	28.4	60.548	28.4	21.738	21.538
36	162.98672	6.4168	12.542	25.4	5996.64	28.484	60.732	28.5	21.9	21.616
37	162.55196	6.4012	12.4924	25.394	6002.94	28.5	60.288	28.5	21.9	21.7
37	163.58616	6.4404	12.552	25.4	5999.84	28.5	60.428	28.472	21.8	21.572
38	162.746	6.42	12.53	25.35	6002	28.5	60.55	28.5	21.9	21.7
38	162.20152	6.404	12.4752	25.328	6002.6	28.5	59.988	28.572	21.872	21.672
39	163.13946	6.4482	12.5752	25.3	5998.06	28.5	60.37	28.594	21.9	21.606
39	163.04624	6.4232	12.5064	25.384	5998.32	28.6	60.236	28.6	21.9	21.7
40	162.679	6.43	12.521	25.3	6001.24	28.6	60.01	28.638	21.8	21.6
40	389.7508	17.216	25.256	22.64	6579.6	28.6	84.3	28.66	21.9	21.7
41	254.1484	10.502	14.6074	24.2	8077.26	28.6	84.318	28.6	21.882	21.6
41	249.4764	10.35	14.4804	24.104	8049.84	28.7	83.7	28.6	21.804	21.604
42	250.5053	10.3626	14.5174	24.174	8027.22	28.7	83.77	28.626	21.926	21.726
42	250.70232	10.3596	14.53	24.2	8016.48	28.7	84.168	28.7	21.848	21.6
43	250.1554	10.337	14.501	24.2	8009.5	28.7	83.56	28.73	21.87	21.6
43	251.10856	10.416	14.5884	24.108	8003.16	28.7	82.864	28.608	21.992	21.7
44	250.43274	10.3914	14.5728	24.1	8004.58	28.7	83.684	28.714	21.9	21.686
44	249.91472	10.3544	14.5352	24.136	8001.92	28.7	83.392	28.736	21.9	21.7
45	248.9771	10.331	14.4684	24.1	8002.74	28.8	83.158	28.758	21.9	21.642
45	249.7724	10.364	14.542	24.1	7999	28.8	83.86	28.8	21.92	21.7
46	250.37008	10.3888	14.5588	24.1	7999.06	28.8	82.542	28.8	21.9	21.7
46	249.79168	10.3648	14.5248	24.1	7998	28.8	83.856	28.724	21.976	21.7
47	249.30486	10.3446	14.4784	24.1	7998.54	28.8	83.884	28.746	21.954	21.754
47	249.68564	10.3604	14.4836	24.1	8000.96	28.8	83.224	28.8	21.932	21.7
48	250.1022	10.382	14.523	24.09	7997.4	28.81	82.78	28.89	21.89	21.6
48	249.55068	10.3548	14.476	24.1	7999.64	28.9	83.98	28.8	22	21.7
49	249.15064	10.3666	14.5098	24.034	8000.96	28.9	83.498	28.9	21.966	21.766
49	249.97484	10.3724	14.5036	24.1	7998.32	28.9	83.528	28.9	21.956	21.656
50	249.34836	10.38	14.51	24.022	8001.12	28.9	83.322	28.9	22	21.722
50	248.64	10.36	14.47	24	8000	28.9	84.3	28.9	22	21.8

Appendix H – TBRe Dyno data obtained from data logger and calculated values.

Measured Torque	Data Logger RPM	VESC RPM	Rad/s	Power	Elec Power	Efficiency	Ideal Torque
0.01	0	0.00	0.00	0.00	0.00	0.00	0.00
0.03	0	0.00	0.00	0.00	0.00	0.00	0.00
0.03	0	0.00	0.00	0.00	0.00	0.00	0.00
0.02	0	0.00	0.00	0.00	0.00	0.00	0.00
0.03	0	0.00	0.00	0.00	0.00	0.00	0.00
0.03	0	0.00	0.00	0.00	0.00	0.00	0.00
0.02	0	0.00	0.00	0.00	0.00	0.00	0.00
0.03	0	0.00	0.00	0.00	0.00	0.00	0.00
0.04	0	0.00	0.00	0.00	0.00	0.00	0.00
0.04	0	0.00	0.00	0.00	0.00	0.00	0.00
0.05	0	0.00	0.00	0.00	0.00	0.00	0.00
0.04	0	0.00	0.00	0.00	0.00	0.00	0.00
0.06	0	0.00	0.00	0.00	0.00	0.00	0.00
0.06	0	0.00	0.00	0.00	0.00	0.00	0.00
0.07	0	0.00	0.00	0.00	0.00	0.00	0.00
0.06	0	0.00	0.00	0.00	0.00	0.00	0.00
0.06	0	0.00	0.00	0.00	0.00	0.00	0.00
0.06	0	0.00	0.00	0.00	0.00	0.00	0.00
0.06	0	0.00	0.00	0.00	0.00	0.00	0.00
0.06	0	0.00	0.00	0.00	0.00	0.00	0.00
0.05	0	0.00	0.00	0.00	0.00	0.00	0.00
4.95	0	0.00	0.00	0.00	0.00	0.00	0.00
1.79	0	129.58	13.57	24.29	90.99	26.69	6.71
2.96	149	152.00	15.92	47.12	37.51	125.60	2.36
2.2	147	148.23	15.52	34.15	35.97	94.94	2.32
1.52	0	147.14	15.41	23.42	34.94	67.02	2.27
1.62	0	144.35	15.12	24.49	36.22	67.61	2.40
2.25	36	143.13	14.99	33.72	37.20	90.66	2.48
2.67	143	143.13	14.99	40.02	36.50	109.63	2.44
2.56	142	144.23	15.10	38.67	35.48	108.97	2.35
2	0	143.54	15.03	30.06	36.54	82.28	2.43
1.12	0	141.85	14.85	16.64	37.68	44.15	2.54
1.61	35	142.80	14.95	24.08	37.97	63.41	2.54
2.58	143	143.77	15.06	38.84	36.50	106.42	2.42
2.81	143	142.76	14.95	42.01	36.70	114.48	2.45
1.41	136	142.33	14.90	21.02	37.94	55.39	2.55
0.95	0	142.99	14.97	14.23	38.32	37.12	2.56
1.71	48	142.71	14.94	25.56	37.55	68.06	2.51
2.84	143	143.34	15.01	42.63	36.99	115.25	2.46
2.65	142	142.98	14.97	39.68	38.03	104.35	2.54
2.2	141	142.14	14.89	32.75	38.84	84.31	2.61
2.44	106	286.97	30.05	73.32	111.94	65.50	3.73
2.6	94	284.73	29.82	77.52	98.82	78.45	3.31
2.65	141	284.19	29.76	78.87	98.68	79.92	3.32
2.44	142	282.59	29.59	72.21	99.66	72.45	3.37
2.56	94	284.41	29.78	76.24	101.37	75.22	3.40
2.59	142	286.47	30.00	77.70	100.34	77.43	3.34
2.64	285	283.14	29.65	78.28	101.12	77.41	3.41

2.77	142	285.85	29.93	82.92	100.48	82.52	3.36
2.74	142	283.34	29.67	81.30	101.97	79.73	3.44
2.71	95	286.29	29.98	81.25	102.30	79.42	3.41
3.02	142	284.32	29.77	89.92	101.90	88.24	3.42
2.54	142	286.85	30.04	76.30	103.43	73.77	3.44
2.55	95	287.80	30.14	76.85	102.47	75.00	3.40
2.47	142	283.58	29.70	73.35	103.59	70.81	3.49
2.27	285	285.25	29.87	67.81	102.84	65.93	3.44
2.61	142	285.45	29.89	78.02	103.04	75.72	3.45
2.22	142	283.42	29.68	65.89	102.68	64.17	3.46
2.79	95	287.31	30.09	83.94	103.05	81.46	3.43
2.88	142	283.50	29.69	85.50	103.95	82.25	3.50
3.17	285	287.19	30.07	95.33	103.58	92.04	3.44
3.33	434	432.58	45.30	150.85	181.25	83.23	4.00
3.37	431	430.95	45.13	152.08	181.51	83.79	4.02
3.38	431	429.83	45.01	152.14	181.08	84.02	4.02
3.39	431	429.15	44.94	152.35	181.69	83.85	4.04
3.34	143	429.07	44.93	150.07	181.57	82.65	4.04
3.32	428	428.96	44.92	149.14	181.27	82.27	4.04
3.34	143	428.87	44.91	150.00	181.27	82.75	4.04
3.3	428	428.36	44.86	148.03	182.58	81.08	4.07
3.31	431	428.54	44.88	148.54	182.74	81.28	4.07
3.33	437	428.47	44.87	149.42	182.21	82.00	4.06
3.36	428	428.42	44.86	150.74	182.33	82.68	4.06
3.36	142	428.33	44.85	150.71	182.78	82.46	4.07
3.38	428	428.78	44.90	151.77	182.43	83.19	4.06
3.35	431	428.56	44.88	150.34	183.55	81.91	4.09
3.32	428	428.71	44.89	149.05	182.97	81.46	4.08
3.27	428	428.76	44.90	146.82	182.51	80.44	4.06
3.34	437	428.43	44.87	149.85	183.77	81.54	4.10
3.31	428	428.45	44.87	148.51	183.70	80.84	4.09
3.31	143	428.66	44.89	148.58	183.90	80.80	4.10
3.57	428	469.97	49.22	175.70	492.38	35.68	10.00
3.75	588	576.95	60.42	226.57	300.36	75.43	4.97
3.99	576	574.99	60.21	240.25	297.05	80.88	4.93
4	576	573.37	60.04	240.17	297.41	80.76	4.95
3.99	566	572.61	59.96	239.25	297.32	80.47	4.96
4	571	572.11	59.91	239.64	296.67	80.78	4.95
4.01	566	571.65	59.86	240.05	298.94	80.30	4.99
4	571	571.76	59.87	239.50	298.23	80.31	4.98
4.01	576	571.57	59.85	240.02	297.17	80.77	4.96
4.01	566	571.62	59.86	240.04	297.53	80.68	4.97
4.01	571	571.36	59.83	239.93	298.48	80.38	4.99
4.01	571	571.36	59.83	239.93	299.20	80.19	5.00
4.01	571	571.29	59.82	239.90	298.51	80.37	4.99
4.01	571	571.32	59.83	239.91	297.92	80.53	4.98
4.01	571	571.50	59.85	239.99	298.38	80.43	4.99
4.01	576	571.24	59.82	239.88	299.11	80.20	5.00
4.02	566	571.40	59.84	240.55	299.25	80.38	5.00
4.02	576	571.50	59.85	240.59	299.59	80.30	5.01
4.02	571	571.31	59.83	240.51	299.76	80.23	5.01
4.02	571	571.51	59.85	240.59	299.98	80.20	5.01

3.3	576	571.43	59.84	197.47	299.40	65.95	5.00

Appendix I – Raw TBRe Dyno data obtained from data logger.

TIME	RPM	LOAD 1	LOAD 2	TORQUE
0	0	-0.04	0.01	0.01
49.9	0	-0.13	0.03	0.03
100	0	-0.12	0.07	0.03
150	0	-0.07	0.07	0.02
200	0	-0.13	0.05	0.03
249.9	0	-0.11	0.07	0.03
300	0	-0.1	0.04	0.02
350	0	-0.12	0.05	0.03
399.9	0	-0.17	0.05	0.04
450	0	-0.15	0.1	0.04
500	0	-0.16	0.14	0.05
550	0	-0.15	0.12	0.04
600	0	-0.19	0.15	0.06
650	0	-0.18	0.21	0.06
699.9	0	-0.19	0.22	0.07
750	0	-0.18	0.19	0.06
800	0	-0.17	0.19	0.06
850	0	-0.2	0.18	0.06
900	0	-0.2	0.16	0.06
950	0	-0.21	0.15	0.06
1000	0	-0.15	0.14	0.05
1050	0	-19.86	10.49	4.95
1099.9	0	-4.3	6.65	1.79
1150	149	-10.05	8.13	2.96
1199.9	147	-10.55	2.91	2.2
1249.9	0	-7.9	1.42	1.52
1299.9	0	-4.86	5.06	1.62
1350	36	-7.09	6.73	2.25
1400	143	-9.57	6.79	2.67
1450	142	-10.96	4.74	2.56
1500	0	-9.64	2.65	2
1550	0	-3.95	2.93	1.12
1599.9	35	-5.21	4.65	1.61
1650	143	-7.57	8.27	2.58
1700	143	-9.97	7.3	2.81
1750	136	-7.19	1.46	1.41
1799.9	0	-4.5	1.35	0.95
1850	48	-4.05	6.43	1.71

1900	143	-9.64	7.75	2.84
1949.9	142	-10.85	5.42	2.65
1999.9	141	-9.82	3.66	2.2
2050	106	-9.07	5.9	2.44
2099.9	94	-8.86	7.07	2.6
2149.9	141	-10.13	6.12	2.65
2200	142	-9.07	5.91	2.44
2250	94	-7.68	7.99	2.56
2300	142	-9.29	6.61	2.59
2350	285	-7.7	8.51	2.64
2400	142	-8.6	8.37	2.77
2449.9	142	-9.97	6.83	2.74
2500	95	-8.35	8.26	2.71
2550	142	-9.93	8.58	3.02
2599.9	142	-9	6.57	2.54
2650	95	-7.57	8.07	2.55
2700	142	-9.18	5.95	2.47
2750	285	-7.57	6.33	2.27
2800	142	-8.57	7.41	2.61
2850	142	-7.57	6.06	2.22
2899.9	95	-8.37	8.74	2.79
2950	142	-9.95	7.73	2.88
3000	285	-8.9	10.56	3.17
3050	434	-11.36	9.07	3.33
3100	431	-11.45	9.23	3.37
3150	431	-11.48	9.25	3.38
3200	431	-11.57	9.21	3.39
3250	143	-11.58	8.93	3.34
3300	428	-11.39	9	3.32
3349.9	143	-11.66	8.82	3.34
3400	428	-11.47	8.75	3.3
3450	431	-11.41	8.9	3.31
3500	437	-11.61	8.8	3.33
3550	428	-11.5	9.12	3.36
3600	142	-11.44	9.14	3.36
3650	428	-11.58	9.18	3.38
3699.9	431	-11.67	8.88	3.35
3749.9	428	-11.42	8.98	3.32
3800	428	-11.37	8.72	3.27
3850	437	-11.71	8.78	3.34
3900	428	-11.44	8.84	3.31
3950	143	-11.55	8.77	3.31
4000	428	-12.52	9.39	3.57

4049.9	588	-12.63	10.36	3.75
4100	576	-13.57	10.91	3.99
4149.9	576	-13.58	10.98	4
4200	566	-13.56	10.95	3.99
4250	571	-13.55	10.98	4
4300	566	-13.59	10.99	4.01
4350	571	-13.56	11.01	4
4400	576	-13.53	11.05	4.01
4450	566	-13.54	11.07	4.01
4499.9	571	-13.54	11.06	4.01
4550	571	-13.53	11.06	4.01
4600	571	-13.48	11.1	4.01
4649.9	571	-13.54	11.08	4.01
4700	571	-13.52	11.06	4.01
4750	576	-13.51	11.11	4.01
4800	566	-13.55	11.13	4.02
4850	576	-13.53	11.12	4.02
4900	571	-13.54	11.11	4.02
4950	571	-13.53	11.1	4.02
5000	576	-12.6	7.64	3.3
5050	148	2.03	-1.29	0.54
5100	0	-0.35	2.23	0.42
5150	0	-0.45	2.18	0.43
5200	0	-0.46	2.13	0.42
5250	0	-0.45	2.11	0.42
5300	0	-0.43	2.09	0.41
5349.9	0	-0.43	2.06	0.41
5400	0	-0.49	2.08	0.42
5449.9	0	-0.51	2.08	0.42
5500	0	-0.51	2.05	0.42
5550	0	-0.56	2.03	0.42
5600	0	-0.56	2.03	0.42
5650	0	-0.58	2.06	0.43
5700	0	-0.58	2.04	0.43
5750	0	-0.65	1.99	0.43
5800	0	-0.63	2	0.43
5850	0	-0.62	1.99	0.43
5900	0	-0.65	2.02	0.44
5950	0	-0.65	2	0.43
5999.9	0	-0.66	1.99	0.43